Jutek : Journal of Usability and Technology Engineering Knowledge

journal homepage: https://ejournal.undar.or.id/index.php/Jutek

Artikel Tinjauan

Studi Implementasi REST API pada Aplikasi Logbook Magang Berbasis Web Menggunakan Laravel

Rizky Parlika [a], Deborah Christian Lewi [b]

[a, b] *Program Studi Informatika, Universitas Pembangunan Nasional "Veteran" Jawa Timur, Surabaya, Indonesia*
email: [a] rizkyparlika.if@upnjatim.ac.id, [b] 23081010050@student.upnjatim.ac.id

INFO ARTICLE

Kata Kunci:
Rest API,
Laravel,
Logbook,
MySQL,
Web

ABSTRAK

Dalam kegiatan harian mahasiswa saat melakukan praktik kerja lapangan(PKL), logbook magang memiliki peran sangat penting guna mencatat setiap kegiatan harian. Namun, sistem pencatatan konvensional memiliki beberapa masalah, seperti merupakan sebuah tugas yang membosankan, kurang aman seperti informasi dapat hilang, dan membuat pelatih kewalahan dengan menyandang tanggung jawab yang besar. Untuk mengatasi kekurangan tersebut, penelitian ini dikembangkan REST API sistem yang dibangun menggunakan kerangka kerja Laravel untuk mendukung pencatatan peserta secara sistematis dan memisahkan antarmuka frontend dan backend. Rest API yang dikembangkan pada penelitian ini memiliki fungsi penambahan data, pengeditan, penghapusan, dan pengambilan data dengen format JSON. Aplikasi web frontend mengakses REST middleware menggunakan metode GET, POST, PUT, dan DELETE. Penelitian yang dilakukan juga menjelaskan mengenai struktur basis data, endpoint, komunikasi antara frontend dan backend. Kesimpulannya, implementasi menunjukkan bahwa aplikasi dapat membantu peserta lebih mudah mencatat dan dengan lebih rapi. Backend juga memungkinkan pengembangan lebih lanjut dan pengintegrasian dengan perangkat mobile altrenatif.

* Corresponding author.
E-mail address: rizkyparlika.if@upnjatim.ac.id (Rizky Parlika).

https://doi.org/xx.xxxx/Jutek.2026.xx.xxx
Received 26 January 2026; Received in revised form 30 January 2026; Accepted 10 February 2026

1. PENDAHULUAN

Magang merupakan langkah awal bagi setiap mahasiswa untuk menimba pengalaman kerja secara langsung. Selain itu, selama menjalani magang, mahasiswa wajib membuat catatan harian melalui logbook yang merupakan bukti kegiatan dan penilaian dalam magang. Oleh karena itu, mahasiswa menginput logbook dalam pencatatan yang terstruktur dan mudah diakses. Pada kenyataannya, banyak instansi yang dalam penyusunan logbook masih menggunakan catatan manual menggunakan buku atau formulir kertas ataupun sudah ada yang memanfaatkan format digital, tetapi terpisah dan tidak ada satu sistem. Penginputan secara manual dapat menimbulkan beberapa masalah, yang mana menjadi risiko hilangnya data, pendataan yang sulit dicari, hingga proses rekapitulasi yang memakan waktu. Penggunaan teknologi web dikembangkan yang memudahkan dalam proses pencatatan logbook secara digital melalui sistem REST API. REST API memisahkan fungsi frontend dan backend sehingga dalam pengembangan sistemnya lebih flexsible dan mudah, sedangkan laravel sendiri merupakan framework PHP yang dilengkapi dengan fitur membangun REST API lebih cepat dan terstruktur.

2. METODOLOGI PENILITIAN

Penelitian ini dimulai dengan studi literatur. Tujuannya jelas: memahami konsep logbook magang, REST API, dan pemanfaatan Laravel sebagai framework pengembangan web. Ketiga aspek ini sudah banyak dibahas dalam penelitian sebelumnya (Herdiyatmoko, 2022a). Setelah itu, peneliti masuk ke tahap analisis kebutuhan, baik fungsional maupun nonfungsional. Fokus utamanya pada pencatatan aktivitas harian mahasiswa dan efisiensi sistem, sesuai rekomendasi dari studi tentang pengembangan sistem informasi digital (Herdiyatmoko, 2022b).

Hasil analisis tadi langsung dituangkan ke dalam rancangan arsitektur sistem, diagram alur, dan desain basis data. Di sini, peneliti benar-benar memperhatikan prinsip kesederhanaan dan efisiensi penyimpanan data. Prinsip-prinsip ini sudah jadi acuan utama dalam berbagai studi tentang perancangan sistem berbasis web dan logbook digital (Widyastuti et al., 2024).

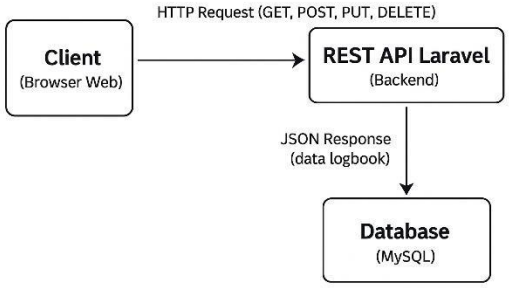
Implementasi dimulai dengan membangun REST API menggunakan Laravel. Basis data utama yang dipakai adalah MySQL. Laravel dipilih bukan tanpa alasan—penelitian sebelumnya menunjukkan framework ini mampu menyederhanakan proses pengembangan backend dan struktur API (Alfath & Ardian, 2024). Setelah REST API selesai, peneliti mengembangkan antarmuka web yang terhubung ke API melalui metode HTTP seperti GET, POST, PUT, dan DELETE.

Langkah selanjutnya, pengujian fungsional. Di tahap ini, peneliti memastikan semua fitur CRUD pada logbook berjalan sesuai kebutuhan pengguna, seperti yang disarankan dalam penelitian evaluasi sistem informasi berbasis web (Ibrahim & Yuridka, 2019). Terakhir, peneliti mengevaluasi hasil implementasi dan menyusun kesimpulan, untuk membuka peluang pengembangan berikutnya.

2.1 Arsitektur Sistem

Arsitektur sistem ini menganut pola client server. Backend Laravel menyediakan endpoint REST API, sementara frontend berperan sebagai klien yang mengakses layanan melalui HTTP. Pembagian tugas seperti ini sudah sejalan dengan model arsitektur sistem modern, seperti yang dijelaskan dalam beberapa penelitian (Sinlae et al., 2024).

Semua data logbook tersimpan di basis data MySQL, lengkap dengan atribut seperti id, nama, tanggal, judul_kegiatan, deskripsi, durasi_jam, dan status. Sistem bertukar data dalam format JSON. Standar ini memang efisien untuk komunikasi data, sesuai pembahasan di literatur (Nurjaman et al., 2024). Gambar 1 menampilkan diagram arsitektur sistem secara visual.



Gambar 1. *Hubungan Client-Server*

2.2 Desain Basis Data dan Endpoint REST API

2.2.1 Desain Basis Data

Desain basis data di sini hanya memakai satu tabel utama, yaitu logbooks. Semua aktivitas harian peserta magang langsung tersimpan di tabel ini. Pendekatan seperti ini memang efisien dan bisa mengurangi redundansi, sejalan dengan praktik perancangan sistem informasi yang sering dibahas di berbagai penelitian (Saputro et al., 2025).

Tabel logbooks ini menampung semua informasi penting soal aktivitas magang, mulai dari identitas pengguna, waktu pelaksanaan, durasi kegiatan, sampai status perkembangan pekerjaan. Struktur tabelnya sudah mengikuti aturan normaliasi dasar, jadi data nggak dobel dan informasi tetap aman. Detail struktur tabelnya bisa dilihat di Tabel 1.

Tabel 1. *Struktur Tabel Logbooks*

No	Nama Kolom	Tipe Data	Keterangan
1	id	BIGINT (Auto Increment)	Primary key adalah identitas unik setiap logbook
2	nama	VARCHAR(100)	Nama lengkap pelaku
3	tanggal	DATE	Tanggal kegiatan
4	judul_kegiatan	VARCHAR(255)	Nama kegiatan
5	deskripsi	TEXT	Penjelasan kegiatan yang dilakukan
6	durasi_jam	INT	Lama waktu kegiatan
7	status	ENUM('belum','proses','selesai')	Status progres aktivitas
8	created_at	TIMESTAMP	Waktu pencatatan data dibuat
9	updated_at	TIMESTAMP	Waktu pembaruan terakhir dari data

Pada MySQL saya pakai struktur ini karena logbook memang nggak butuh relasi yang ribet. Sistem ini cuma perlu memastikan setiap aktivitas tercatat dan bisa dipantau dengan jelas. Kolom-kolom yang ada juga udah pas—misalnya kolom status, yang bikin kita gampang lihat progres pekerjaan tanpa ribet.

2.2.2 Desain Endpoint REST API

REST API (Representational State Transfer Application Programming Interface) berperan sebagai penghubung antara frontend dan server. API ini mengikuti prinsip RESTful, jadi aksesnya simpel, rapi, dan fleksibel—bisa dipakai di berbagai platform, nggak keikat sama satu teknologi aja. Endpoint yang saya buat di sini khusus untuk operasi CRUD—Create, Read, Update, Delete—di data logbook. Berikut ini daftar lengkap endpoint yang bisa langsung kamu gunakan.

Tabel 2. *Daftar Endpoint Rest Api Logbook Magang*

No	Endpoint	Method	Deskripsi Fungsi

1	/api/logbook	GET	Mengambil seluruh daftar logbook dari database
2	/api/logbook/{id}	GET	Mengambil detail satu logbook berdasarkan ID
3	/api/logbook	POST	Menambahkan data logbook baru
4	/api/logbook/{id}	PUT	Memperbarui data logbook berdasarkan ID
5	/api/logbook/{id}	DELETE	Menghapus data logbook berdasarkan ID

REST API yang tersedia antara lain:

A. GET /api/logbook = Menampilkan semua data logbook (Hadi et al., 2020)
B. GET /api/logbook/{id} = Menampilkan detail satu entri logbook
C. POST /api/logbook = Menambah entri baru ke dalam logbook (Widia & Firmansyah, 2022)
D. PUT /api/logbook/{id} = Memperbarui entri berdasarkan ID (Alfath & Ardian, 2024)
E. DELETE /api/logbook/{id} = Menghapus entri logbook tertentu (Herdiyatmoko, 2022a)

2.3 Alur Komunikasi Sistem (REST API Flow)

Berikut adalah penjelasan naratif mengenai alur komunikasi antara pengguna, frontend, dan server:

A. Pengguna mengisi formulir logbook → data dikirim ke frontend.
B. Frontend mengirim request ke server menggunakan fetch API (POST/PUT/GET/DELETE).
C. Server memvalidasi request dan berinteraksi dengan basis data.
D. Server mengirimkan response JSON berisi status, pesan, dan data logbook.
E. Frontend menampilkan data ke pengguna, baik dalam tabel maupun modal detail.

Narasi ini menggambarkan hubungan siklus data antara pengguna dan server sesuai prinsip REST.

3. HASIL DAN PEMBAHASAN

3.1 Implementasi Backend REST API

Saya memilih Laravel untuk membangun backend aplikasi ini karena framework ini punya struktur yang jelas model, controller, dan rute dipisahkan rapi. Di bagian model, entitas Logbook langsung dikaitkan dengan tabel logbooks. Ini bikin pemetaan data jadi lebih cepat dan efisien. Semua logika inti aplikasi ditempatkan di LogbookController. Di sana, operasi-operasi utama seperti menampilkan semua data, menambah entri baru, memperbarui isi logbook, sampai menghapus catatan yang tak diperlukan, semuanya diatur. Pendekatan ini mengikuti pola REST API: format respons konsisten, HTTP method digunakan dengan benar, komunikasi antara klien dan server jadi lebih simple (Shofiana et al., 2024).

app > Http > Controllers > Api > LogbookController.php

```php
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use App\Models\Logbook;
use Illuminate\Http\Request;

class LogbookController extends Controller
{
    // GET /api/logbook
    public function index()
    {
        $logbooks = Logbook::orderBy('tanggal', 'desc')->get();

        return response()->json([
            'success' => true,
            'data'    => $logbooks,
        ]);
    }

    // GET /api/logbook/{id}
    public function show($id)
    {
        $logbook = Logbook::findOrFail($id);

        return response()->json([
            'success' => true,
            'data'    => $logbook,
        ]);
    }
```

Gambar 2. *Logbook Controller Method index() dan show()*

Fungsi index() dan show() jadi kunci untuk membaca data. index() mengambil semua entri dari database lalu mengurutkannya dari yang paling baru. show() menampilkan detail satu entri tertentu. Laravel punya findOrFail(), jadi kalau data yang dicari tidak ada, API langsung mengirim respons error otomatis. Fitur ini membantu sistem tetap andal, sejalan dengan konsep API modern (Kusuma et al., 2020). Kedua fungsi ini jadi pondasi supaya frontend bisa menampilkan daftar logbook dan detailnya secara dinamis.

```php
    // POST /api/logbook
    public function store(Request $request)
    {
        $data = $request->validate([
            'nama'           => 'required|string|max:100',
            'tanggal'        => 'required|date',
            'judul_kegiatan' => 'required|string|max:255',
            'deskripsi'      => 'nullable|string',
            'durasi_jam'     => 'required|integer|min:1',
            'status'         => 'required|in:belum,proses,selesai',
        ]);

        $logbook = Logbook::create($data);

        return response()->json([
            'success' => true,
            'message' => 'Logbook berhasil dibuat',
            'data'    => $logbook,
        ]);
    }

    // PUT /api/logbook/{id}
    public function update(Request $request, $id)
    {
        $data = $request->validate([
            'nama'           => 'required|string|max:100',
            'tanggal'        => 'required|date',
            'judul_kegiatan' => 'required|string|max:255',
            'deskripsi'      => 'nullable|string',
            'durasi_jam'     => 'required|integer|min:1',
            'status'         => 'required|in:belum,proses,selesai',
        ]);

        $logbook = Logbook::findOrFail($id);
        $logbook->update($data);
```

Gambar 3. *Logbook Controller Method POST dan PUT: store & update*

Untuk menambah atau memperbarui data, saya pakai store() dan update(). Di kedua fungsi ini, validasi input diterapkan ketat—format tanggal dicek, panjang judul diperiksa, status kegiatan divalidasi. Validasi ini penting, supaya integritas data terjaga dan tidak ada data aneh yang lolos ke database (Puspitaningrum et al., 2022). Setelah semua validasi beres, data disimpan atau diperbarui lewat Eloquent ORM, lalu API mengirimkan respons JSON sebagai tanda kalau permintaan sudah dijalankan dengan sukses.

```
68
69          return response()->json([
70              'success' => true,
71              'message' => 'Logbook berhasil diperbarui',
72              'data'    => $logbook,
73          ]);
74      }
75
76      // DELETE /api/logbook/{id}
77      public function destroy($id)
78      {
79          $logbook = Logbook::findOrFail($id);
80          $logbook->delete();
81
82          return response()->json([
83              'success' => true,
84              'message' => 'Logbook berhasil dihapus',
85          ]);
86      }
87  }
88
```

Gambar 4. *Logbook Controller Method DELETE: destroy*

Metode destroy() berfungsi untuk menghapus entri tertentu di tabel logbooks. Penghapusan berbasis ID memastikan hanya data yang benar-benar diinginkan pengguna yang terhapus. Respons JSON menjaga komunikasi tetap sederhana dan mudah dipahami frontend, sejalan dengan prinsip REST API yang memang jadi standar di banyak sistem pelaporan dan aktivitas harian (Kumar et al., 2021).

Endpoint GET, POST, PUT, dan DELETE mengatur alur data antara frontend dan backend. Format JSON memudahkan serialisasi dan membuat data bisa dipakai di berbagai platform tanpa masalah. Pendekatan seperti ini juga membuka jalan untuk pengembangan lebih lanjut, misalnya integrasi ke aplikasi mobile atau sistem manajemen magang digital, seperti yang dijelaskan dalam penelitian tentang logbook elektronik (Agil et al., 2021).

app > Models > Logbook.php

```
1   <?php
2
3   namespace App\Models;
4
5   use Illuminate\Database\Eloquent\Factories\HasFactory;
6   use Illuminate\Database\Eloquent\Model;
7
8   class Logbook extends Model
9   {
10      use HasFactory;
11
12      protected $fillable = [
13          'nama',
14          'tanggal',
15          'judul_kegiatan',
16          'deskripsi',
17          'durasi_jam',
18          'status',
19      ];
20  }
21
```

Gambar 5. *Model Logbook.php*

Model Logbook langsung merepresentasikan tabel logbooks di basis data. Model ini pakai fitur HasFactory dan turunan dari kelas Model di Laravel, jadi proses bikin instance dan kelola data terasa jauh lebih efisien. Di atribut $fillable, ada daftar kolom yang bisa diisi secara massal: nama, tanggal, judul_kegiatan, deskripsi, durasi_jam, dan status. Mekanisme ini nggak cuma ngeringkas proses input, tapi juga bikin sistem lebih aman karena hanya kolom-kolom tertentu yang bisa diisi lewat permintaan API. Cara kerja seperti ini sejalan dengan konsep pengelolaan data yang aman dan terstruktur, seperti yang dijelaskan dalam penelitian soal pengembangan aplikasi Laravel dan sistem logbook digital (Putra et al., 2025).

routes > web.php

```php
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\Api\LogbookController;

// Halaman web utama (UI logbook)
Route::get('/', function () {
    return view('logbook.index');
});

// REST API untuk logbook
Route::prefix('api')->group(function () {
    Route::get('/logbook', [LogbookController::class, 'index']);
    Route::post('/logbook', [LogbookController::class, 'store']);
    Route::put('/logbook/{id}', [LogbookController::class, 'update']);
    Route::delete('/logbook/{id}', [LogbookController::class, 'destroy']);
});

```

Gambar 6. *web.php*

Lanjut ke file web.php. Di sinilah semua rute aplikasi dikonfigurasi. Rute utama, yaitu /, menampilkan halaman antarmuka logbook pengguna bisa nambah atau lihat data di sini. Rute REST API dikumpulkan dengan prefix api/ dan terdiri dari empat operasi inti: GET untuk ambil data, POST buat nambah data, PUT untuk update entri tertentu, dan DELETE buat hapus data berdasarkan ID. Semua rute ini dikaitkan ke metode yang pas di LogbookController. Dengan struktur rute yang jelas dan rapi, komunikasi antara frontend dan backend jadi lancar, dan standar REST API tetap terjaga seperti yang banyak dipakai di sistem informasi modern (Maulana et al., 2021).

routes > api.php

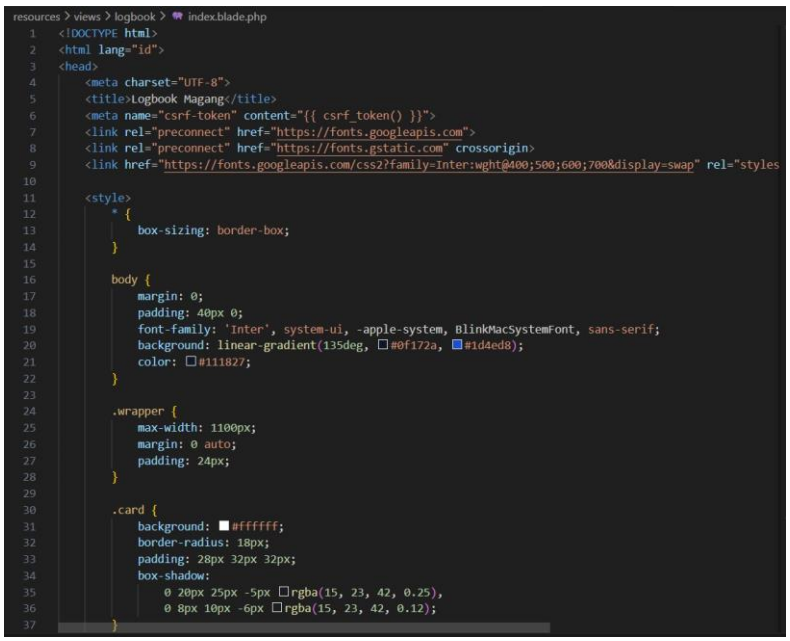
```php
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\Api\LogbookController;

Route::get('/logbook', [LogbookController::class, 'index']);
Route::get('/logbook/{id}', [LogbookController::class, 'show']);
Route::post('/logbook', [LogbookController::class, 'store']);
Route::put('/logbook/{id}', [LogbookController::class, 'update']);
Route::delete('/logbook/{id}', [LogbookController::class, 'destroy']);

```

Gambar 7. *api.php*

Sementara itu, file api.php memuat semua endpoint REST API yang aplikasi gunakan buat akses data logbook. Endpoint GET /logbook menjalankan index() buat ngambil semua data logbook, dan GET /logbook/{id} menjalankan show() untuk detail entri tertentu. Untuk tambah data, ada POST /logbook yang diproses store(). Update data jalan lewat PUT /logbook/{id}, dan hapus data pakai DELETE /logbook/{id}. Setiap endpoint udah disusun sesuai kaidah REST API, jadi operasi CRUD punya rute yang jelas dan terpisah. Ini bikin integrasi dengan frontend atau sistem eksternal jadi lebih gampang. Pendekatan kayak gini juga selaras dengan praktik pengembangan REST API di berbagai penelitian sebelumnya (Mukaromah et al., 2018).

3.2 Implementasi di Frontend Web

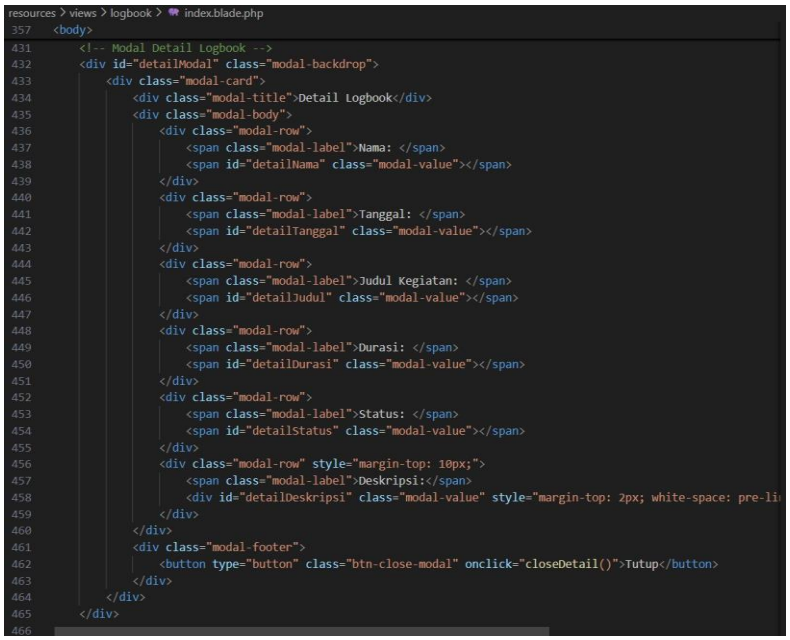
Aplikasi ini menggunakan Blade template untuk membangun antarmuka pengguna, jadi yang pengguna lihat adalah halaman web yang tampak statis. Tapi, semua interaksi datanya mulai dari mengisi formulir sampai menyimpan data sebenarnya dijalankan oleh JavaScript yang langsung berkomunikasi dengan REST API. Kalau pengguna ingin membuat atau mengedit entri, mereka cukup mengisi nama, tanggal, durasi, judul kegiatan, status, dan deskripsi di formulir. Begitu tombol “Simpan” ditekan, JavaScript mengubah semua input itu jadi data JSON, lalu mengirimnya ke endpoint REST API dengan metode fetch pakai POST untuk data baru, atau PUT untuk pembaruan. Cara kerja seperti ini penting untuk menjaga arsitektur aplikasi tetap modular dan konsisten dengan prinsip REST, seperti yang sering dibahas dalam berbagai penelitian tentang pentingnya memisahkan logic dan antarmuka di aplikasi modern (Minarno et al., 2021).



```
resources > views > logbook > index.blade.php
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <title>Logbook Magang</title>
    <meta name="csrf-token" content="{{ csrf_token() }}">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;500;600;700&display=swap" rel="styles

    <style>
        * {
            box-sizing: border-box;
        }

        body {
            margin: 0;
            padding: 40px 0;
            font-family: 'Inter', system-ui, -apple-system, BlinkMacSystemFont, sans-serif;
            background: linear-gradient(135deg, #0f172a, #1d4ed8);
            color: #111827;
        }

        .wrapper {
            max-width: 1100px;
            margin: 0 auto;
            padding: 24px;
        }

        .card {
            background: #ffffff;
            border-radius: 18px;
            padding: 28px 32px 32px;
            box-shadow:
                0 20px 25px -5px rgba(15, 23, 42, 0.25),
                0 8px 10px -6px rgba(15, 23, 42, 0.12);
        }
```

Gambar 8. *index.blade.php (1)*

Di sisi tampilan, file index.blade.php jadi pusat dari seluruh UI. Di sana, ada pengaturan metadata, integrasi font dari Google Fonts, sampai token CSRF yang memastikan setiap permintaan HTTP tetap aman. Desain visualnya diatur dengan CSS internal; di sini warna, layout, dan tampilan komponen dibuat cukup detail. Latar linear gradient warna biru bikin kesan modern, sementara class .wrapper membantu supaya tata letaknya tetap rapi dan terpusat. Komponen .card jadi wadah utama: tampil putih bersih, sudutnya membulat, efek bayangannya halus jadinya kelihatan profesional. Pendekatan desain seperti ini memang sejalan dengan prinsip UI/UX dalam pengembangan aplikasi logbook dan web lain yang mengutamakan kemudahan serta keterbacaan (Putra et al., 2025). Data logbook sendiri ditampilkan dalam tabel yang rapi. Ini membantu pengguna menemukan dan meninjau kembali informasi dengan mudah—jelas, terstruktur, dan tidak membingungkan (Syahidin & Ramadhan, 2021).



```
resources > views > logbook > index.blade.php
357  <body>
431      <!-- Modal Detail Logbook -->
432      <div id="detailModal" class="modal-backdrop">
433          <div class="modal-card">
434              <div class="modal-title">Detail Logbook</div>
435              <div class="modal-body">
436                  <div class="modal-row">
437                      <span class="modal-label">Nama: </span>
438                      <span id="detailNama" class="modal-value"></span>
439                  </div>
440                  <div class="modal-row">
441                      <span class="modal-label">Tanggal: </span>
442                      <span id="detailTanggal" class="modal-value"></span>
443                  </div>
444                  <div class="modal-row">
445                      <span class="modal-label">Judul Kegiatan: </span>
446                      <span id="detailJudul" class="modal-value"></span>
447                  </div>
448                  <div class="modal-row">
449                      <span class="modal-label">Durasi: </span>
450                      <span id="detailDurasi" class="modal-value"></span>
451                  </div>
452                  <div class="modal-row">
453                      <span class="modal-label">Status: </span>
454                      <span id="detailStatus" class="modal-value"></span>
455                  </div>
456                  <div class="modal-row" style="margin-top: 10px;">
457                      <span class="modal-label">Deskripsi:</span>
458                      <div id="detailDeskripsi" class="modal-value" style="margin-top: 2px; white-space: pre-li
459                  </div>
460              </div>
461              <div class="modal-footer">
462                  <button type="button" class="btn-close-modal" onclick="closeDetail()">Tutup</button>
463              </div>
464          </div>
465      </div>
466
```

Gambar 9. *index.blade.php (2)*

Model Logbook jadi penghubung langsung antara aplikasi dan tabel logbooks di database. Dengan fitur bawaan Laravel seperti HasFactory dan Eloquent ORM, ngatur data dan bikin objek terasa jauh lebih mudah. Properti $fillable juga nggak kalah penting. Lewat pengaturan ini, kita tentuin kolom mana aja yang bisa diisi

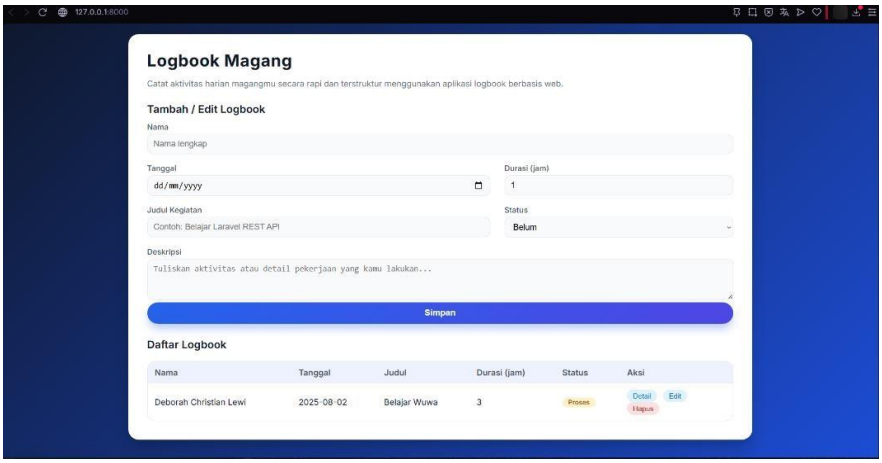
massal, jadi lebih aman dari manipulasi data yang nggak valid. Cara kerja seperti ini sesuai banget sama praktik pengembangan sistem modern struktur backend jadi rapi, aman, dan gampang dikelola, apalagi kalau dipakai buat layanan REST API.

REST API dan Laravel sama-sama menekankan betapa pentingnya struktur backend yang aman dan terkontrol (Nurjaman et al., 2024), juga penggunaan model sebagai inti arsitektur aplikasi (Herfandi & Julkarnain, 2022; Ula et al., 2024).

3.3 Tampilan Antarmuka

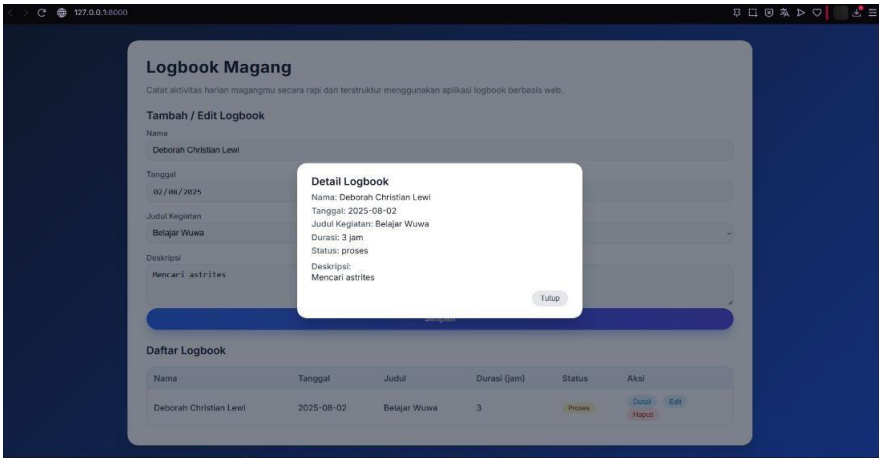
Kalau dibandingkan dengan logbook manual, aplikasi ini jelas punya banyak keunggulan. Data tersimpan di satu tempat, cari informasi juga jauh lebih cepat, dan aksesnya gampang—cukup lewat browser tanpa perlu instal apa-apa lagi. Backend-nya dipisah sebagai REST API, jadi nanti gampang kalau mau bikin klien lain, entah itu aplikasi mobile atau apa pun (Chebotov et al., 2021).

Tapi, masih ada kekurangan. Sampai sekarang, belum ada autentikasi pengguna. Fitur pelaporan atau dashboard buat pembimbing juga belum tersedia. Soal performa dan keamanan? Jujur saja, belum dianalisis lebih dalam. Dua hal ini bakal jadi fokus penting buat pengembangan ke depannya (Adrian et al., 2021).



Gambar 10. *Tampilan Logbook (1)*

Aplikasi logbook ini pakai HTML, CSS, sama Blade Laravel buat tampilannya. Hasilnya, tampilannya modern dan gampang dipahami. Form inputnya disusun dengan grid, jadi nama, tanggal, durasi, judul, status, dan deskripsi kelihatan rapi, tetap responsif di semua perangkat juga. Latar belakangnya gradasi biru, ada card putih di tengah supaya mata langsung fokus ke bagian input data. Tombol “Simpan”, “Detail”, “Edit”, sama “Hapus” warnanya beda-beda, jadi langsung kelihatan mana yang mana. Data logbook ditampilkan di tabel simpel, statusnya pakai badge warna biar gampang kelihatan. Desain kayak gini bikin ngisi logbook jadi jauh lebih cepat, nyaman, dan efisien (Veronika Butarbutar, 2022).



Gambar 11. *Tampilan Logbook (2)*

3.4 Tabel Perbandingan

Tabel 3. *Perbandingan Penilitian Terkait Aplikasi Logbook/ Pencatatan Aktivitas*

Peneliti & Tahun	Objek/ Studi Kasus	Teknologi yang Digunakan	Kelebihan	Kekurangan
Sari (2021)	Aplikasi logbook mahasiswa magang	PHP Native & MySQL	Sistem sederhana, mudah digunakan, dan cepat dikembangkan	Tidak mendukung API, tampilan kurang
Putra & Handayani (2022)	Logbook digital berbasis mobile	Flutter + Firebase	Real-time sync, UI lebih interaktif, mendukung mobile	Tidak tersedia versi web untuk admin, kurang fleksibel untuk integrasi sistem lain
Wijaya (2023)	Sistem manajemen aktivitas harian pegawai	Laravel + Blade	Struktur backend kuat, keamanan lebih baik, mendukung CRUD penuh	Tidak menyediakan REST API, hanya berjalan untuk web
Penelitian Ini / Sistem Usulan (2025)	Logbook magang berbasis web dengan REST API	Laravel REST API + Blade Frontend	Mendukung CRUD via API, UI modern, responsif, mudah diintegrasikan	Belum memiliki autentikasi user dan belum mendukung mobile app

Perbandingan penelitian terkait ditunjukkan pada Tabel 3, yang menyoroti perbedaan objek, teknologi, serta kekuatan dan kelemahan masing–masing penelitian (Hartono et al., 2025).

Tabel 4. *Perbedaan dan Persamaan Sistem Usulan Dengan Penilitian Terkait*

Aspek	Penelitian Terdahulu	Sistem Usulan (Aplikasi Logbook REST API)	Persamaan
Platform	Banyak penelitian hanya berbasis web atau mobile saja	Berbasis web dengan REST API yang dapat diperluas ke mobile	Sama-sama menyediakan fitur pencatatan aktivitas
Teknologi Backend	PHP Native / Laravel tanpa API / Firebase	Laravel REST API full CRUD	Sama-sama menggunakan database relasional
Fitur Logbook	Umumnya hanya input & lihat data	Input, edit, delete, detail modal popup	Fitur inti pencatatan tetap ada
Tampilan (UI/UX)	Beberapa tampilan masih sederhana	Desain modern, responsif, memakai gradient & card UI	Mengutamakan kemudahan penggunaan
Integrasi Sistem	Kurang mendukung integrasi	API memungkinkan integrasi aplikasi mobile / web lain	Masih mendukung ekspansi ke fitur tambahan

Tabel 4 menunjukkan persamaan dan perbedaan sistem usulan dengan penelitian terdahulu dari berbagai aspek, termasuk teknologi, fitur, dan tampilan (Mohidin, 2022).

Tabel 5. *Potensi Penilitian Masa Depan*

Aspek	Penelitian Terdahulu	Sistem Usulan (Aplikasi Logbook REST API)
Autentikasi Pengguna	Sistem belum memiliki login khusus user	Penambahan role admin & peserta magang (JWT / Sanctum)

Mobile App	Belum tersedia aplikasi Android/iOS	Pengembangan aplikasi Flutter yang terhubung ke REST API
Analisis Aktivitas	Belum ada grafik atau statistik otomatis	Menambah dashboard analitik (chart durasi mingguan/bulanan)
Ekspor Data	Tidak ada fitur ekspor laporan	Tambah fitur export PDF/Excel otomatis
Notifikasi	Tidak ada pengingat aktivitas	Integrasi notifikasi email/WhatsApp untuk pengingat logbook harian

Tabel 5 menjelaskan potensi pengembangan di masa depan berdasarkan keterbatasan sistem yang ada saat ini (Rahmat et al., 2025).

4. KESIMPULAN

Penelitian ini berhasil membangun REST API untuk aplikasi logbook magang berbasis web dengan framework Laravel. Sistemnya bisa mengelola entri logbook—mulai dari menambah, memperbarui, menghapus, sampai menampilkan data dalam format JSON. Antarmuka web yang dibuat juga sudah pakai REST API ini, jadi peserta magang bisa lebih mudah mencatat aktivitas harian mereka.

Waktu diuji, aplikasi ini bisa menyimpan data seperti nama, tanggal, judul kegiatan, durasi, status, dan deskripsi aktivitas dengan rapi. Fitur daftar logbook dan tampilan detailnya bikin pengguna gampang meninjau kembali aktivitas yang sudah mereka lakukan. Karena frontend dan backend terpisah lewat REST API, sistem ini juga terbuka buat integrasi dengan aplikasi lain ke depannya.

Untuk pengembangan berikutnya, ada beberapa saran. Misalnya, menambah modul autentikasi dan otorisasi pengguna, mengintegrasikan sistem ini dengan sistem informasi akademik supaya data logbook bisa langsung terhubung ke penilaian magang, dan mengembangkan aplikasi mobile yang pakai REST API yang sama. Pengujian performa dan keamanan juga penting, supaya sistem tetap aman dan lancar kalau dipakai lebih banyak orang.

DAFTAR PUSTAKA

Adrian, Q. J., Devija, R. N., & Setiawansyah. (2021). Penerapan Sistem Informasi Administrasi Perpustakaan Menggunakan Model Desain User Experience. *Jurnal Manajemen Informatika (JAMIKA)*, *11*(1). https://doi.org/10.34010/JAMIKA.V11I1

Agil, M. M., Kartika, R., Diyah, R., Bahri, S., & Nurhalifah, S. (2021). Pemanfaatan Google Spreadsheet Sebagai Media Penyimpanan Data Masyarakat Rw. 04 Kp. Cilayung. *Proceedings UIN Sunan Gunung Djati Bandung*, *1*. https://proceedings.uinsgd.ac.id/index.php/proceedings/article/download/771/690

Alfath, M., & Ardian, F. (2024). Implementasi REST API Aplikasi Pendaftaran Siswa Baru di SMA Hikmah Yapis Papua Berbasis Android. *Jurnal Indonesia: Manajemen Informatika Dan Komunikasi*, *5*, 445–453. https://doi.org/10.35870/jimik.v5i1.507

Chebotov, A. I., Gertsenberger, K. V., Slepov, I. P., & Moshkin, A. A. (2021). Electronic Logbook platform for NICA experiments. *AIP Conf. Proc*, *2377*. https://doi.org/10.1063/5.0063399

Hadi, R., Defni, & Medrofa, A. C. (2020). Designing an Online Supervision System (Logbook) with Laravel Framework. *International Journal of Advanced Science Computing and Engineering*, *2*(2), 69–75. https://doi.org/10.62527/ijasce.2.2.49

Hartono, F. Z., Umam, K., Hakim, L., Prasetyo, J. A., & Febrita, R. E. (2025). Perbandingan Performa Framework Laravel dengan ExpressJS Pada Pengembangan Aplikasi Homestay Kosasih. *Jikom: Jurnal Informatika Dan Komputer*, *15*(1). http://ojs.stikombanyuwangi.ac.id/index.php/jikom/article/view/158

Herdiyatmoko, H. F. (2022a). Back-End System Design Based on Rest Api. *Jurnal Tekinkom (Teknik*

Informasi Dan Komputer), *5*(1), 123. https://doi.org/10.37600/tekinkom.v5i1.401

Herdiyatmoko, H. F. (2022b). Desain sistem backend berbasis REST API menggunakan Framework Laravel 7. *SKANIKA: Sistem Komputer Dan Teknik Informatika*, *5*(2), 136–144. https://doi.org/10.36080/skanika.v5i2.2947

Herfandi, M. H., & Julkarnain, M. (2022). Desain dan Implementasi Restful Web Services untuk Integrasi Data dan Aplikasi. *Jurnal Informatika Teknologi Dan Sains (Jinteks)*, *4*(1), 36–41. https://doi.org/10.51401

Ibrahim, & Yuridka, F. (2019). APLIKASI LOGBOOK BERBASISKAN PEMPROGRAMAN PHP DAN MYSQL PADA METRO LINK BANJARMASIN DI PT. INDOSAT DAN XL. *JAZARI: JURNAL ILMIAH TEKNIK MESIN*, *4*(1). https://doi.org/10.31602/al-jazari.v4i1.1960

Kumar, A., Haider, Y., Kumar, M., & Khan, R. (2021). Using WhatsApp as a quick-access personal logbook for maintaining clinical records and follow-up of orthopedic patients. *Cureus*, *13*(1), 1–6. https://doi.org/10.7759/cureus.12900

Kusuma, W. A., Ghufron, K. M., & Fauzan. (2020). Penggunaan User Persona Untuk Evaluasi Dan Meningkatkan Ekspektasi Pengguna Dalam Kebutuhan Sistem Informasi Akademik. *SINTECH (Science and Information Technology) Journal*, *2*(1), 1–10. http://ejournal.instiki.ac.id/index.php/sintechjournal/article/view/587

Maulana, R., Sulistyanto, A., & Rini, A. S. (2021). Perancangan sistem informasi pengajuan dan pelaporan pembayaran tunjangan kinerja pada lembaga pemasyarakatan salemba berbasis web menggunakan. *Jurnal Manajamen Informatika Jayakarta*, *1*(4), 283. https://doi.org/10.52362/jmijayakarta.v1i4.507

Minarno, A. E., Syafa'ah, L., & Rahayu, D. (2021). Science and Technology for Community: Improving Web Programming Skills using Laravel Framework. *Jurnal Dedikasi*, *18*(1), 21–26. https://doi.org/10.22219/dedikasi.v18.i1.15590

Mohidin, I. K. (2022). Penerapan Teknologi Rest Api pada Aplikasi Perpustakaan Digital Politeknik Gorontalo. *Jurnal Technopreneur (JTech)*, *10*(1), 34–39. https://doi.org/10.30869/jtech.v10i1.922

Mukaromah, S., Kusumantara, P. M., Putra, A. B., & Pratama, A. (2018). Analysis and Design Logbook Information Systems. *Atl. Press*, *1*, 1039–1044. https://doi.org/10.2991/icst-18.2018.210

Nurjaman, I., Utomo, F. S., & Hermanto, N. (2024). Penerapan REST API Laravel sebagai Fondasi Back-end Aplikasi G-MOOC 4D. *Journal of Informatics and Interactive Technology*, *1*(1), 9–18. https://doi.org/10.63547/jiite.v1i1.4

Puspitaningrum, P. N. E. D., Hariyati, T. S., & Muhaeriwati, T. (2022). *P. N. Elisabeth Dewi Puspitaningrum, Rr. Tutik Sri... - Google Scholar*. J. Keperawatan Silampari. https://scholar.google.com/scholar?hl=id&as_sdt=0%2C5&q=P.+N.+Elisabeth+Dewi+Puspitaningrum%2C+Rr.+Tutik+Sri+Hariyati%2C+Titiek+Muhaeriwati%2C+"PENGGUNAAN+E-LOGBOOK+PRECEPTORSHIP+UNTUK+MEMPERMUDAH+PROGRAM+PRECEPTORSHIP+PERAWAT+BARU+DI+RS+X+JAKARTA%3A+PROG

Putra, F. P. E., Efendi, R. W., Tamam, A. B., & Pramadi, W. A. (2025). Tren dan Praktik Terbaik dalam Pengembangan Web Berbasis API: Kajian Literatur terhadap Framework Laravel dan React. *Infomatek*, *27*(1), 165–178. https://doi.org/10.23969/infomatek.v27i1.25122

Rahmat, E. S. F., Al Haritsi, S., & Anjasmara, R. M. (2025). *Perancangan Aplikasi Pengelolaan Logbook Magang Berbasis Web Di Badan Pusat Statistik Provinsi Riau | Rahmat | Journal of Accounting Law Communication and Technology*. JALAKOTEK. https://doi.org/https://doi.org/10.57235/jalakotek.v2i2.6349

Saputro, M. I., Pertiwi, S., Setiadi, D., & Sopian, A. (2025). Science and Technology for the Community Training in Building Professional REST API Services using Laravel at SMKN 64 East Jakarta. *Urnal Pemberdayaan Komunitas MH Thamrin*, *7*(1), 81–92. https://doi.org/10.37012/jpkmht.v7i1.2564

Shofiana, D. A., Mardiansyah, S. S., Parabi, M. I., & Syarif, A. (2024). Implementasi REST API Menggunakan JSON WEB Token (JWT) Pada Sistem Monitoring KPI Berbasis Mobile di PT Industri Kereta API (Persero). *Jurnal Komputasi*, *12*(2), 101–111. https://doi.org/10.23960/komputasi.v12i2.272

Sinlae, F., Irwanda, E., Maulana, Z., & Eka Syahputra, V. (2024). Penggunaan Framework Laravel

dalam Membangun Aplikasi Website Berbasis PHP. *J. Siber Multi Disiplin*, *2*(2), 119–132. https://doi.org/10.38035/jsmd.v2i2.186

Syahidin, Y., & Ramadhan, R. (2021). REST API Architecture Design on Multi-Platform Device Development. *Jurnal E-Komtek (Elektro-Komputer-Teknik)*, *5*(2), 178–189. https://doi.org/10.37339/e-komtek.v5i2.762

Ula, M. I., Kusrini, K., & Muhammad, A. H. (2024). PENERAPAN PRINSIP-PRINSIP RESTFUL API DALAM PENGEMBANGAN WEB SERVICE PADA NEO FEEDER DENGAN FRAMEWORK LARAVEL: PENERAPAN. *Nusantara of Engineering (NOE)*, *7*(1). https://ojs.unpkediri.ac.id/index.php/noe/article/view/21417

Veronika Butarbutar, S. (2022). E-Logbook Sebagai Aktivitas Pembuktian Kegiatan Klinis Perawat Melalui Rekam Medik Elektronik: Tinjauan Literatur. *Jurnal Ilmu Kesehatan Insan Sehat*, *10*(1). https://pdfs.semanticscholar.org/1052/a50181433f241d7ef8caed451062b8949f06.pdf

Widia, I. D. M., & Firmansyah, A. (2022). Rancang Bangun Aplikasi Pengelolaan Aktivitas Proyek (Logbook App) Berbasis Android Di Pt. Inka Madiun (Persero). *VOK@SINDO J. Ilmu-Ilmu Terap. Dan Has. Karya Nyata*, *undefined*(2), 51. https://doi.org/10.21776/ub.vokasindo.2021.009.2.4

Widyastuti, I., Harike, M. H., Takbir, M. N., Malik, A., & Sari, J. Y. (2024). Digital Transformation of Libraries: Web-based Information System Development with Laravel. *Journal of Embedded Systems, Security and Intelligent Systems*, 147–152. https://doi.org/10.59562/jessi.v5i2.3330