

BAB III

PELAKSANAAN KERJA MAGANG

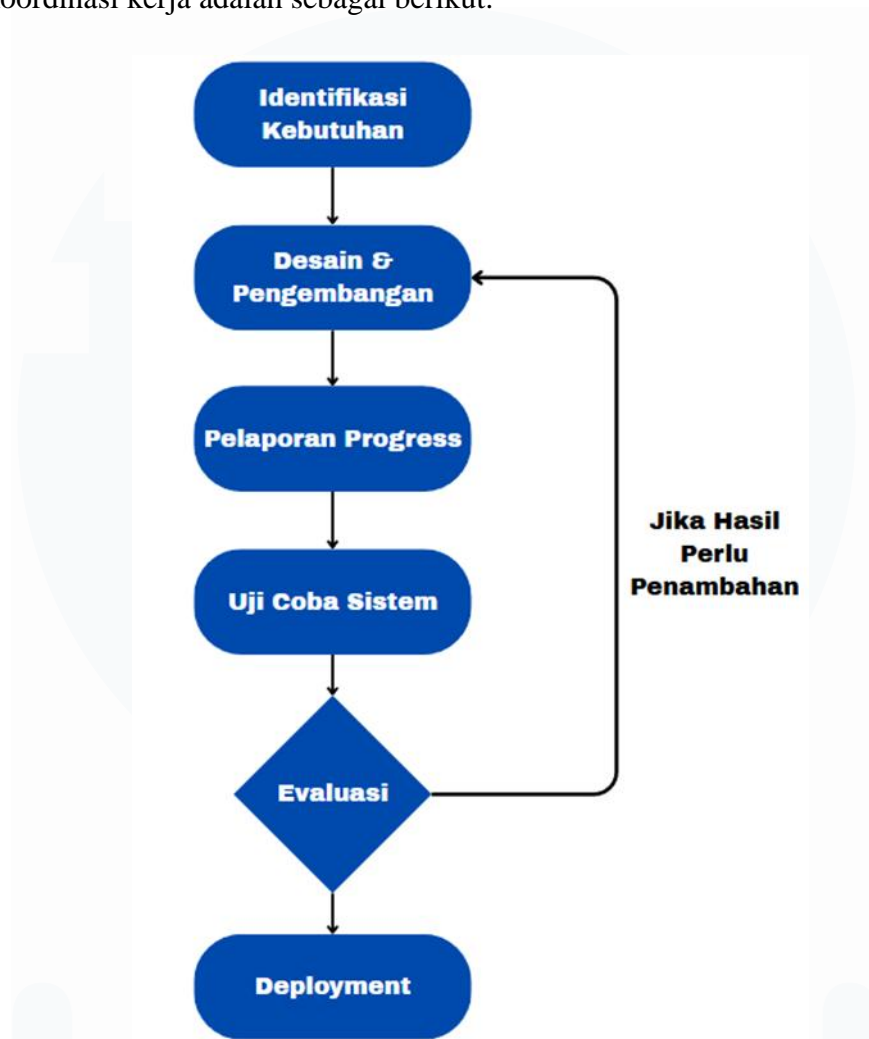
3.1 Kedudukan dan Koordinasi

Dalam pelaksanaan kerja magang ini, saya berada di bawah supervisor langsung pemilik perusahaan yang juga merupakan penerjemah tersumpah serta satu-satunya pengelola usaha. Mengingat perusahaan berstatus auto-entrepreneur di Prancis, struktur organisasi bersifat sederhana, sehingga seluruh proses komunikasi, arahan pekerjaan, hingga evaluasi hasil kerja dilakukan secara langsung antara saya dan supervisor tanpa adanya hierarki internal lainnya.

Sebagai peserta magang, saya berperan sebagai *web developer* yang bertanggung jawab dalam merancang dan membangun situs web untuk mendukung operasional layanan penerjemahan dan interpretasi milik supervisor. Kedudukan saya secara langsung berada di bawah supervisi pemilik usaha, sehingga setiap keputusan teknis, kebutuhan fitur, revisi desain, maupun penyesuaian konten dikonsultasikan langsung kepada beliau.

Peran ini memberi saya ruang untuk bekerja secara mandiri, sekaligus mengharuskan saya memahami kebutuhan bisnis, standar profesional penerjemah tersumpah, serta proses komunikasi lintas negara dalam konteks layanan resmi.

Alur koordinasi kerja adalah sebagai berikut:



Gambar 3.1 Alur Kerja

Seluruh proses koordinasi selama masa magang dilakukan secara online, mengingat supervisor berdomisili di Prancis dan seluruh pekerjaan dikerjakan secara remote. Pola komunikasi yang diterapkan mencakup:

- Meeting mingguan yang dijadwalkan apabila memungkinkan, digunakan untuk melaporkan progres, menerima masukan, dan membahas kebutuhan teknis atau desain baru.

- Komunikasi harian atau sewaktu diperlukan melalui pesan singkat atau email untuk klarifikasi teknis, permintaan revisi, dan koordinasi kecil terkait konten.
- Pengiriman hasil kerja secara bertahap, seperti prototipe halaman website, rancangan alur formulir, atau hasil implementasi backend, yang kemudian dievaluasi oleh supervisor.
- Umpan balik langsung diberikan oleh supervisor untuk memastikan seluruh proses pengembangan sesuai dengan standar profesional yang ingin ditampilkan melalui situs web.

Pola koordinasi yang fleksibel dan intensif ini memungkinkan proses pengembangan berjalan efektif meskipun dilakukan sepenuhnya secara daring dengan perbedaan zona waktu antara Indonesia dan Prancis. Selain itu, alur komunikasi yang sederhana dan langsung mempermudah proses diskusi, pengambilan keputusan, dan penyesuaian kebutuhan proyek secara cepat.

3.2 Tugas dan Uraian Kerja Magang

Selama masa magang, telah dikerjakan beberapa proyek yang telah ditugaskan. Rincian proyek yang dikerjakan dapat dilihat pada tabel berikut:

Tabel 3.2 Timeline Kerja Magang

Bulan	Minggu	Deskripsi
September	1	Perkenalan dengan Supervisor dan pendiskusan proyek
September	2	Brainstorm dan experimenting tech stack untuk proyek
September	3	Pembuatan database dan mailing system di backend
September	4	Menyempurnakan mailing system dari feedback supervisor, serta uji coba dengan provider mailer dari Brevo
October	1	Pembuatan Frontend

October	2	Modifikasi design Frontend dari feedback Supervisor
October	3	Modifikasi design Frontend dari feedback Supervisor
October	4	Integrasi Frontend dengan Backend
October	5	Bugfixing kedua Frontend dan Backend
November	1	Modifikasi hasil Frontend dari feedback Supervisor
November	2	Mencari dan Experimenting provider paling murah untuk me-deploy frontend dan backend
November	3	Backend deployment menggunakan provider Koyeb
November	4	Frontend deployment menggunakan provider Vercel
December	1	Modifikasi kecil bagian frontend, post-deployment

Landasan Teori:

Dalam melaksanakan tugas selama masa magang ini, saya menggunakan beberapa landasan teori yang berkaitan langsung dengan pengembangan website modern. Pengembangan sisi backend dilakukan menggunakan Laravel Framework versi 12, yaitu salah satu framework PHP yang paling banyak digunakan untuk membangun aplikasi web berskala kecil hingga besar. Laravel menyediakan arsitektur MVC, sistem routing, Eloquent ORM, middleware, serta fitur keamanan bawaan yang memudahkan proses pengembangan layanan API yang stabil dan terstruktur. Framework ini sangat cocok digunakan untuk layanan backend yang memerlukan pengiriman email, pengelolaan request dari frontend, serta manajemen data yang terpusat.

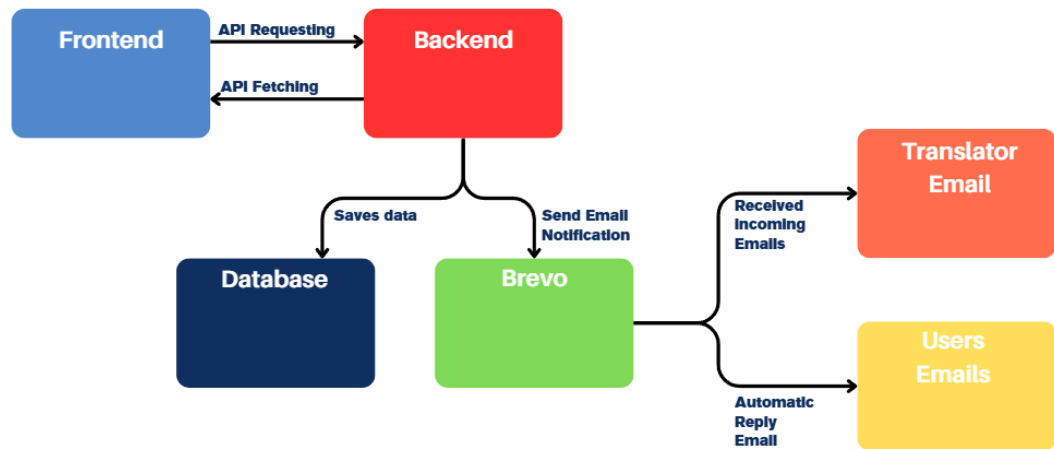
Di sisi frontend, saya menggunakan React dengan Typescript, yang merupakan library JavaScript modern berbasis komponen. React memungkinkan pembuatan antarmuka pengguna yang interaktif, responsif, dan mudah dikembangkan melalui sistem komponen yang dapat digunakan kembali. Penggunaan Typescript memberikan manfaat tambahan berupa type-checking sehingga error dapat terdeteksi sejak proses development, bukan ketika aplikasi sudah berjalan.

Untuk kebutuhan penyimpanan data, perusahaan menggunakan PostgreSQL, yaitu sistem manajemen basis data relasional yang terkenal stabil, aman, dan mendukung operasi kompleks. Meskipun proyek ini hanya memerlukan satu tabel saja, penggunaan PostgreSQL tetap memberikan kelebihan dari sisi integritas data dan kemudahan integrasi dengan Laravel melalui fitur Eloquent ORM.

Seluruh proses pengiriman email resmi dari website ke inbox penerjemah dilakukan melalui Brevo, sebuah email delivery service berbasis API yang mendukung pengiriman email dengan tingkat deliverability yang tinggi. Integrasi Brevo dengan Laravel dilakukan menggunakan API key sehingga backend dapat mengirim email secara otomatis berdasarkan permintaan pengguna di frontend.

Untuk keperluan deployment, backend Laravel dipublikasikan menggunakan platform cloud Koyeb, yang menyediakan lingkungan serverless dengan kemampuan auto-deployment serta integrasi Git. Sementara itu, bagian frontend direkomendasikan untuk di-deploy melalui Vercel, platform hosting khusus untuk aplikasi frontend modern seperti React, karena memiliki performa tinggi dan proses deployment yang sangat mudah.

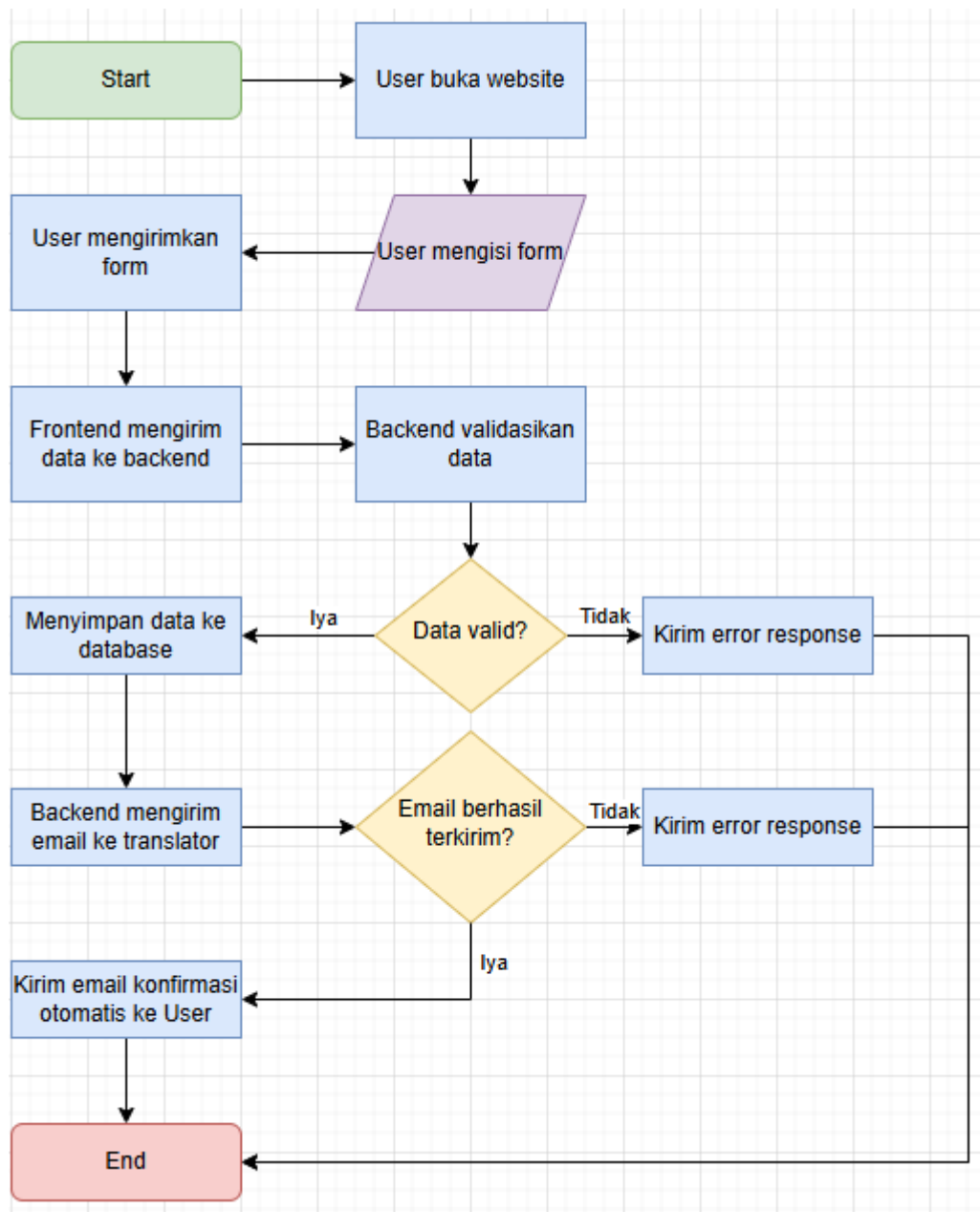
Agar aplikasi dapat diakses secara publik dengan identitas profesional, domain resmi dibeli melalui Hostinger dan dikonfigurasi untuk mengarahkan trafik baik ke backend di Koyeb maupun ke frontend di Vercel. Seluruh komponen teknologi ini bekerja secara terintegrasi sehingga aplikasi dapat berjalan dengan efisien, aman, dan mudah digunakan oleh klien.



Gambar 3.2 System Architecture

1. Pengguna mengakses website melalui browser untuk melihat informasi layanan penerjemahan serta mengisi formulir permintaan penerjemahan secara online.
2. Frontend website berfungsi sebagai antarmuka pengguna, yang menampilkan formulir permintaan, menerima input data (nama, email, nomor telepon, catatan, dan dokumen), serta mengirimkan data tersebut ke backend melalui permintaan HTTP.
3. Backend API bertugas memproses seluruh permintaan dari frontend, termasuk:
 - Melakukan validasi data yang dikirimkan oleh pengguna.
 - Mengelola unggahan dokumen yang dikirimkan melalui formulir.
 - Menyimpan data permintaan klien ke dalam basis data sebagai arsip permintaan penerjemahan.
4. Basis data digunakan untuk menyimpan informasi permintaan penerjemahan, seperti identitas klien, detail kontak, catatan tambahan, serta lokasi penyimpanan dokumen yang diunggah. Data ini berfungsi sebagai dokumentasi dan referensi bagi penerjemah.
5. Setelah data berhasil diproses dan disimpan, backend akan menjalankan proses pengiriman email otomatis, yaitu:

- Mengirimkan email notifikasi ke email penerjemah sebagai pemberitahuan adanya permintaan penerjemahan baru beserta dokumen yang dilampirkan.
 - Mengirimkan email balasan otomatis (auto-reply) kepada pengguna sebagai konfirmasi bahwa permintaan penerjemahan telah berhasil diterima dan diteruskan kepada penerjemah.
6. Mekanisme pengiriman email otomatis ini membantu memastikan bahwa:
- Permintaan klien tidak terlewat.
 - Komunikasi antara klien dan penerjemah menjadi lebih cepat dan terorganisir.
 - Pengguna mendapatkan kepastian bahwa permintaan mereka telah diterima oleh sistem.



Gambar 3.3 Flowchart proses system

1. Proses dimulai ketika pengguna membuka website dan mengakses halaman formulir permintaan penerjemahan.
2. Pengguna mengisi data yang diperlukan, seperti nama, alamat email, nomor telepon, catatan tambahan, serta mengunggah dokumen yang akan diterjemahkan.

3. Setelah seluruh data diisi, pengguna menekan tombol Submit, dan frontend mengirimkan data tersebut ke backend melalui API dalam bentuk permintaan HTTP.
4. Backend menerima permintaan tersebut dan melakukan validasi data, untuk memastikan seluruh input yang dikirimkan telah sesuai dengan ketentuan sistem.
5. Apabila data tidak valid, backend akan mengembalikan respons error ke frontend, dan sistem menampilkan pesan kesalahan kepada pengguna agar data dapat diperbaiki.
6. Jika data valid, backend akan menyimpan dokumen yang diunggah ke media penyimpanan server.
7. Selanjutnya, backend menyimpan informasi permintaan penerjemahan ke dalam basis data sebagai arsip permintaan klien.
8. Setelah data berhasil disimpan, backend secara otomatis mengirimkan email notifikasi kepada penerjemah untuk memberitahukan adanya permintaan penerjemahan baru beserta dokumen terkait.
9. Pada saat yang sama, sistem juga mengirimkan email balasan otomatis (auto-reply) kepada pengguna sebagai konfirmasi bahwa permintaan telah berhasil diterima dan diteruskan ke penerjemah.
10. Jika email tidak berhasil terkirim, backend akan mengembalikan respons error ke frontend, dan sistem menampilkan pesan bahwa email gagal terkirim.
11. Frontend kemudian menampilkan pesan keberhasilan kepada pengguna sebagai tanda bahwa proses pengiriman permintaan telah selesai.
12. Proses berakhir setelah pengguna menerima konfirmasi di halaman website dan melalui email.

Web Features:

1. Halaman Informasi Layanan

- Website menyediakan halaman informasi yang berisi penjelasan mengenai layanan penerjemahan dan interpretasi yang ditawarkan. Informasi ini mencakup jenis layanan, bahasa yang didukung, serta gambaran umum proses penerjemahan dokumen. Fitur ini bertujuan untuk membantu calon klien memahami layanan yang tersedia sebelum melakukan pengajuan permintaan.

2. Formulir Permintaan Penerjemahan (Submission Form)

- Website menyediakan formulir online yang memungkinkan calon klien untuk mengajukan permintaan penerjemahan secara langsung. Pada formulir ini, pengguna diminta untuk mengisi data identitas seperti nama, alamat email, dan nomor telepon, serta dapat menambahkan catatan tambahan terkait kebutuhan penerjemahan. Selain itu, juga dapat mengunggah dokumen yang akan diterjemahkan melalui sistem.

3. Unggah Dokumen Digital

- Fitur unggah dokumen memungkinkan pengguna untuk mengirimkan berkas yang akan diterjemahkan langsung melalui website. Fitur ini menggantikan proses pengiriman dokumen secara manual melalui email, sehingga lebih praktis dan terstruktur.

4. Validasi Data Input

- Sistem dilengkapi dengan mekanisme validasi data untuk memastikan seluruh informasi yang dikirimkan oleh pengguna telah sesuai dengan format yang ditentukan. Validasi ini dilakukan sebelum data diproses lebih lanjut oleh sistem backend, sehingga dapat mengurangi kesalahan data dan meningkatkan keandalan sistem.

5. Pengiriman Email Notifikasi ke Penerjemah

- Setelah formulir berhasil dikirim dan data dinyatakan valid, sistem secara otomatis mengirimkan email notifikasi kepada penerjemah. Email ini berisi

informasi permintaan penerjemahan beserta dokumen yang diunggah oleh pengguna, sehingga penerjemah dapat segera menindaklanjuti permintaan tersebut.

6. Email Konfirmasi Otomatis kepada Pengguna

- Sistem mengirimkan email balasan otomatis (auto-reply) yang menandakan bahwa permintaan penerjemahan telah berhasil diterima. Fitur ini meningkatkan kepercayaan pengguna serta memberikan kepastian bahwa dokumen telah sampai kepada penerjemah.

7. Penyimpanan Data Permintaan

- Setiap permintaan penerjemahan yang masuk disimpan ke dalam basis data. Penyimpanan ini berfungsi sebagai arsip permintaan klien yang dapat digunakan untuk keperluan pencatatan, penelusuran kembali, atau pengembangan sistem di masa mendatang.

8. Tampilan Responsif

- Website dirancang dengan tampilan yang responsif sehingga dapat diakses dengan baik melalui berbagai perangkat, seperti komputer, tablet, maupun smartphone. Hal ini bertujuan untuk memberikan pengalaman pengguna yang nyaman tanpa bergantung pada jenis perangkat yang digunakan.

Pengembangan Backend Sistem:

Pengembangan backend pada proyek website layanan penerjemah tersumpah ini dilakukan menggunakan Laravel Framework versi 12 dengan PostgreSQL sebagai sistem manajemen basis data. Backend berperan sebagai penghubung utama antara frontend dan layanan eksternal, khususnya untuk memproses data formulir dari pengguna serta mengirimkan email secara otomatis ke inbox penerjemah melalui layanan Brevo.

Secara umum, backend dirancang menggunakan arsitektur client–server dengan pendekatan RESTful API. Frontend hanya bertugas mengirimkan data permintaan, sedangkan seluruh proses validasi, pengolahan data, dan pengiriman email ditangani sepenuhnya oleh backend. Pendekatan ini dipilih untuk menjaga keamanan data pengguna serta memisahkan logika bisnis dari antarmuka pengguna.

```
1 <?php
2
3 use Illuminate\Http\Request;
4 use Illuminate\Support\Facades\Route;
5 use App\Http\Controllers\SubmissionController;
6
7 Route::middleware(middleware: ['auth:sanctum'])->get(uri: '/user', action: function (Request $request): mixed {
8     return $request->user();
9 });
10
11 Route::post(uri: '/submissions', action: [SubmissionController::class, 'store']);
```

Gambar 3.4 Konfigurasi Endpoint API

Backend menyediakan sebuah endpoint utama yang digunakan untuk menerima data formulir dari frontend, yaitu endpoint POST /submissions. Endpoint ini didefinisikan pada file konfigurasi API Laravel dan diarahkan ke sebuah controller khusus yang menangani proses penyimpanan data serta pengiriman email. Dengan pemisahan routing dan controller, sistem menjadi lebih terstruktur dan mudah dikembangkan di kemudian hari.

```

1 reference | 0 overrides
public function store(Request $request): JsonResponse
{
    $request->validate(rules: [
        'name' => 'required',
        'email' => 'required|email',
        'phone_number' => 'required|string|max:20',
        'file' => 'required|file|mimes:pdf,doc,docx',
    ]);

    $filePath = $request->file(key: 'file')->store(path: 'submissions', options: 'public');

    $submission = Submission::create(attributes: [
        'name' => $request->name,
        'email' => $request->email,
        'phone_number' => $request->phone_number,
        'notes' => $request->notes,
        'file_path' => $filePath,
    ]);

    Mail::to(users: 'sguerande@yahoo.fr')
    ->cc(users: 'jonathanryan2015@gmail.com')
    ->send(mailable: new SubmissionReceived(submission: $submission));

    return response()->json(data: [
        'message' => 'Submission sent to translator!',
        'data' => $submission
    ], status: 201);
}

```

Gambar 3.5 Proses Penyimpanan Data dan Pengiriman Email

Kode pada Gambar 3.5 menunjukkan proses utama backend dalam menangani permintaan penerjemahan yang dikirimkan oleh pengguna melalui website. Ketika endpoint diakses, sistem memulai dengan melakukan validasi terhadap data yang diterima untuk memastikan seluruh informasi wajib telah diisi dengan benar, termasuk nama, alamat email, nomor telepon, serta dokumen yang diunggah. Selain memastikan kelengkapan data, backend juga membatasi tipe file yang diperbolehkan agar hanya menerima format dokumen tertentu, sehingga dapat mengurangi risiko keamanan dan kesalahan unggahan.

Setelah data dinyatakan valid, sistem menyimpan dokumen yang diunggah ke dalam direktori penyimpanan publik Laravel dan mencatat lokasi file tersebut. Selanjutnya, data permintaan pengguna beserta path file disimpan ke dalam basis data menggunakan Eloquent ORM, yang mempermudah pengelolaan dan interaksi

dengan basis data PostgreSQL. Setelah proses penyimpanan berhasil, backend secara otomatis mengirimkan email notifikasi kepada penerjemah tersumpah yang berisi informasi pengajuan klien serta dokumen sebagai lampiran. Pada tahap akhir, sistem juga mengirimkan email konfirmasi otomatis kepada pengguna sebagai tanda bahwa permintaan telah berhasil diterima, kemudian mengembalikan respons API dalam format JSON kepada frontend.



```

1 reference | 0 overrides
public function __construct(Submission $submission)
{
    $this->submission = $submission;
}

/**
 * Get the message envelope.
 */
0 references | 0 overrides
public function envelope(): Envelope
{
    return new Envelope(
        subject: 'New Translation Request',
    );
}

/**
 * Get the message content definition.
 */
0 references | 0 overrides
public function content(): Content
{
    return new Content(
        view: 'emails.submission',
        with: [
            'submission' => $this->submission,
        ],
    );
}

/**
 * Get the attachments for the message.
 *
 * @return array<int, \Illuminate\Mail\Mailables\Attachment>
 */
0 references | 0 overrides
public function attachments(): array
{
    return [
        \Illuminate\Mail\Mailables\Attachment::fromPath(
            path: storage_path(path: "app/public/{$this->submission->file_path}")
        ),
    ];
}

```

Gambar 3.6 Implementasi Kelas “SubmissionReceived” sebagai Mailable Laravel

Kode pada Gambar 3.6 menampilkan implementasi kelas Mailable pada Laravel yang digunakan untuk mengatur pengiriman email notifikasi kepada penerjemah. Pada konstruktor kelas, objek data permintaan (Submission) diterima dan disimpan sebagai properti agar dapat digunakan di seluruh bagian email. Sistem

kemudian mendefinisikan subjek email melalui metode `envelope()` serta menentukan tampilan isi email menggunakan template yang telah disiapkan, sehingga informasi permintaan klien dapat disajikan secara terstruktur dan mudah dipahami.

Selain isi email, kelas ini juga mengatur mekanisme penyertaan dokumen sebagai lampiran. Dokumen yang diunggah oleh pengguna diambil langsung dari lokasi penyimpanan server berdasarkan path file yang tersimpan di basis data. Dengan mekanisme ini, setiap email yang dikirimkan kepada penerjemah selalu disertai dokumen yang relevan, sehingga penerjemah dapat langsung mengakses dan meninjau berkas tanpa perlu membuka sistem tambahan. Implementasi ini membantu meningkatkan efisiensi proses kerja serta memastikan seluruh informasi penting terkirim dalam satu alur komunikasi.

```
MAIL_MAILER=smtp
# MAIL_SCHEME=smtp.gmail.com
MAIL_HOST=smtp-relay.brevo.com
MAIL_PORT=587
MAIL_USERNAME=[REDACTED]
MAIL_PASSWORD=[REDACTED]
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS="noreply@sophieguerande.com"
MAIL_FROM_NAME="TranslationWeb"
```

Gambar 3.7 Konfigurasi Layanan Email SMTP Menggunakan Brevo

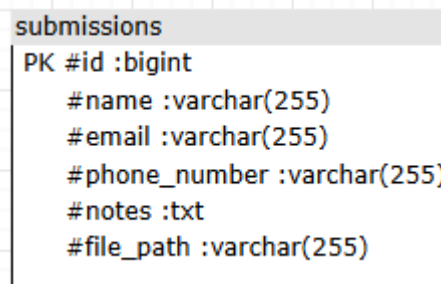
Gambar 3.7 menunjukkan konfigurasi layanan email yang digunakan oleh sistem backend untuk menangani pengiriman email notifikasi secara otomatis. Pada implementasi ini, backend memanfaatkan layanan email transaksional Brevo yang diintegrasikan melalui protokol SMTP. Penggunaan layanan email khusus ini bertujuan untuk memastikan proses pengiriman email berjalan lebih stabil, cepat, serta memiliki tingkat keberhasilan yang tinggi dibandingkan dengan penggunaan layanan email pribadi.

Konfigurasi dilakukan melalui file `environment` Laravel dengan mendefinisikan beberapa parameter utama, seperti jenis mailer yang digunakan (`smtp`), alamat

server SMTP Brevo, port komunikasi, serta mekanisme enkripsi yang digunakan. Port 587 dengan enkripsi TLS dipilih karena merupakan standar komunikasi yang aman untuk pengiriman email, sehingga data yang dikirimkan antara aplikasi dan server email terlindungi dari potensi penyadapan.

Selain itu, sistem menggunakan kredensial autentikasi berupa username dan password yang diberikan oleh Brevo. Seluruh informasi sensitif tersebut disimpan dalam environment variable dan tidak dituliskan secara langsung di dalam kode program. Pendekatan ini bertujuan untuk meningkatkan keamanan aplikasi serta mempermudah pengelolaan konfigurasi, terutama ketika sistem dijalankan pada lingkungan yang berbeda seperti pengembangan lokal dan server produksi.

Pengaturan alamat pengirim (MAIL_FROM_ADDRESS) dan nama pengirim (MAIL_FROM_NAME) digunakan untuk memastikan setiap email yang dikirimkan memiliki identitas yang jelas dan profesional. Dengan konfigurasi ini, seluruh email notifikasi yang dihasilkan oleh sistem backend, baik untuk penerjemah maupun pengguna, dapat terkirim secara konsisten dan terkelola dengan baik sebagai bagian dari alur komunikasi digital pada website.



submissions
PK #id :bigint
#name :varchar(255)
#email :varchar(255)
#phone_number :varchar(255)
#notes :txt
#file_path :varchar(255)

Gambar 3.8 Tabel Submissions

Untuk kebutuhan penyimpanan data, digunakan PostgreSQL, meskipun hanya satu tabel yang dibutuhkan. Tabel ini berfungsi untuk menyimpan data permintaan klien atau metadata tertentu, seperti informasi pengirim dan waktu pengiriman. Penggunaan basis data tetap dipertahankan untuk memastikan data dapat dicatat

secara terstruktur serta memungkinkan pengembangan fitur lanjutan di masa mendatang.

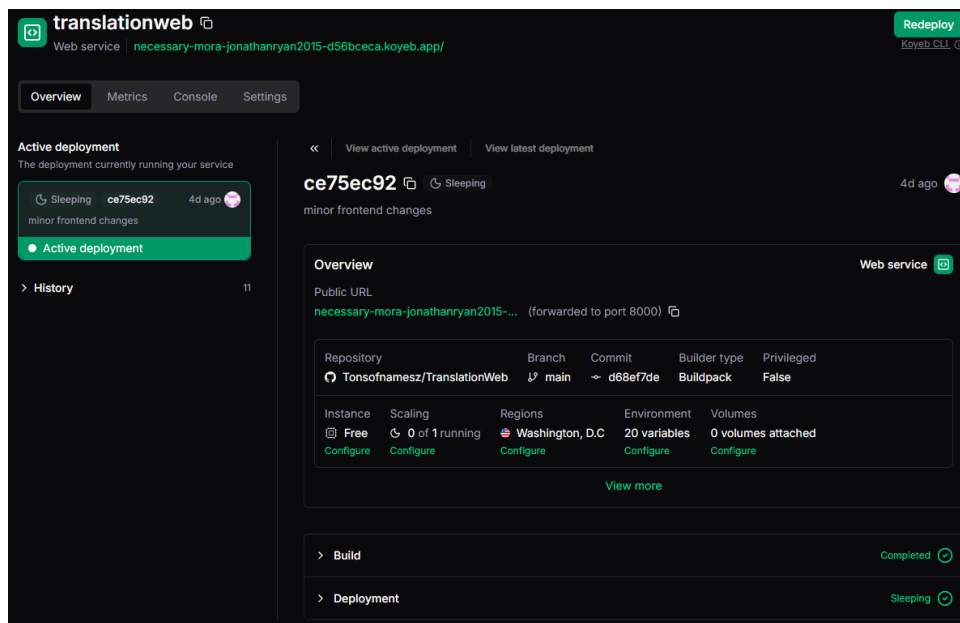
Dalam sistem ini, basis data digunakan untuk menyimpan data permintaan layanan penerjemahan yang dikirimkan oleh pengguna melalui formulir website. Meskipun sistem backend berfokus pada pengiriman email, penyimpanan data ke dalam basis data tetap diperlukan untuk keperluan pencatatan, dokumentasi, serta potensi pengembangan fitur di masa mendatang. Sistem basis data yang digunakan adalah PostgreSQL, dan hanya terdapat satu tabel utama, yaitu tabel submissions.

Tabel submissions dirancang untuk menyimpan informasi penting terkait setiap permintaan yang masuk. Kolom id digunakan sebagai primary key yang bersifat unik untuk membedakan setiap data pengajuan. Kolom ini dibuat secara otomatis oleh sistem untuk memudahkan identifikasi data.

Kolom name digunakan untuk menyimpan nama lengkap pengguna yang mengajukan permintaan penerjemahan. Informasi ini diperlukan agar penerjemah dapat mengetahui identitas pengirim secara jelas. Kolom email berfungsi untuk menyimpan alamat email pengguna, yang dapat digunakan sebagai media komunikasi lanjutan atau konfirmasi terkait permintaan tersebut.

Kolom phone_number disediakan untuk menyimpan nomor telepon pengguna. Keberadaan kolom ini bertujuan untuk memberikan alternatif kontak apabila diperlukan komunikasi lanjutan yang lebih cepat atau bersifat mendesak. Kolom notes bersifat opsional dan digunakan untuk menyimpan keterangan tambahan yang disampaikan oleh pengguna, misalnya jenis dokumen, tenggat waktu, atau kebutuhan khusus lainnya.

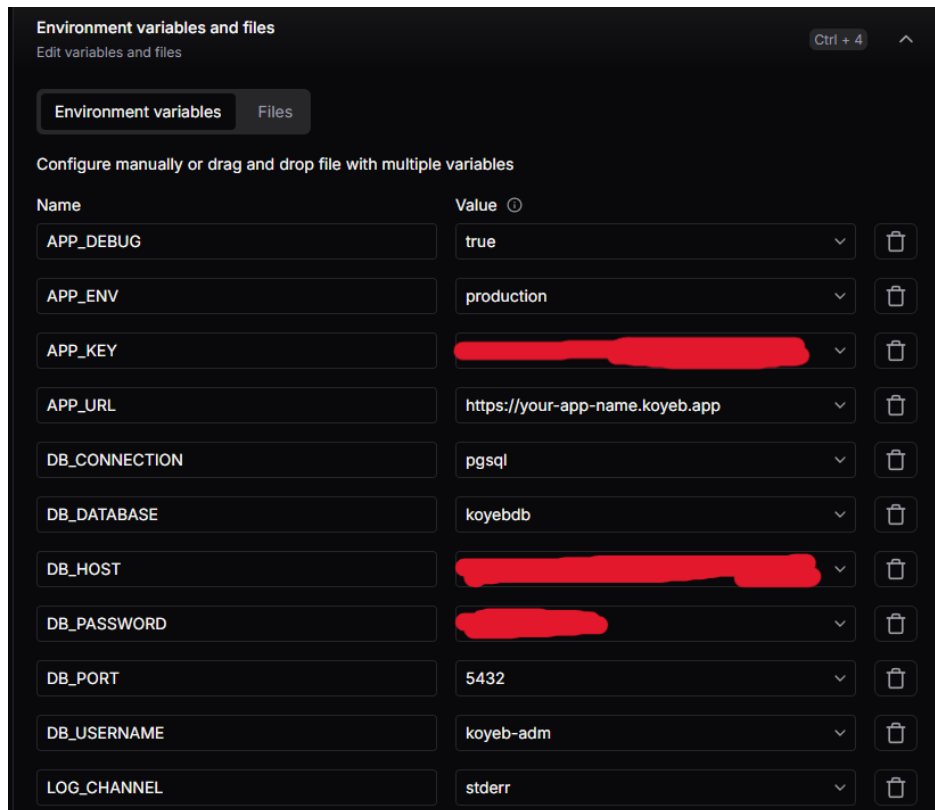
Kolom file_path digunakan untuk menyimpan lokasi file dokumen yang diunggah oleh pengguna ke server. Sistem tidak menyimpan file secara langsung di basis data, melainkan hanya menyimpan path file untuk menjaga efisiensi penyimpanan dan mempermudah pengelolaan file. Dokumen ini kemudian dilampirkan secara otomatis pada email yang dikirimkan ke penerjemah.



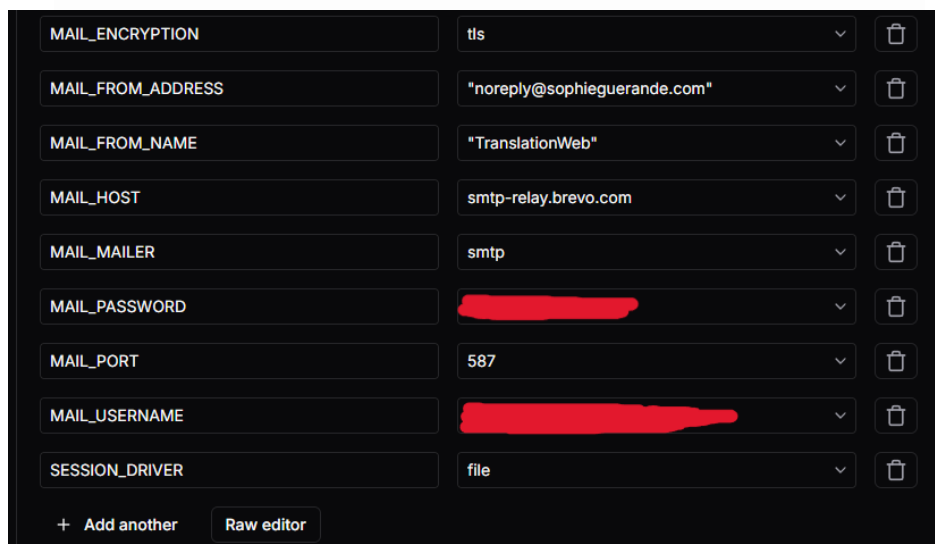
Gambar 3.9 Tampilan Dashboard Koyeb untuk Monitoring Status Backend

Setelah proses pengembangan dan pengujian backend selesai dilakukan di lingkungan lokal, tahap selanjutnya adalah melakukan deployment agar aplikasi dapat diakses secara publik. Backend Laravel dideploy menggunakan platform cloud Koyeb, yang mendukung aplikasi berbasis web dan menyediakan kemudahan dalam pengelolaan environment serta integrasi dengan repository.

Proses deployment dimulai dengan mengunggah kode sumber backend ke repository dan menghubungkannya dengan layanan Koyeb. Pada tahap ini, Koyeb dikonfigurasi untuk menjalankan aplikasi Laravel dengan menyesuaikan kebutuhan runtime, termasuk pengaturan versi PHP dan perintah build yang diperlukan.



Gambar 3.10 Konfigurasi env di Koyeb untuk Backend bagian 1



Gambar 3.11 Konfigurasi env di Koyeb untuk Backend bagian 2

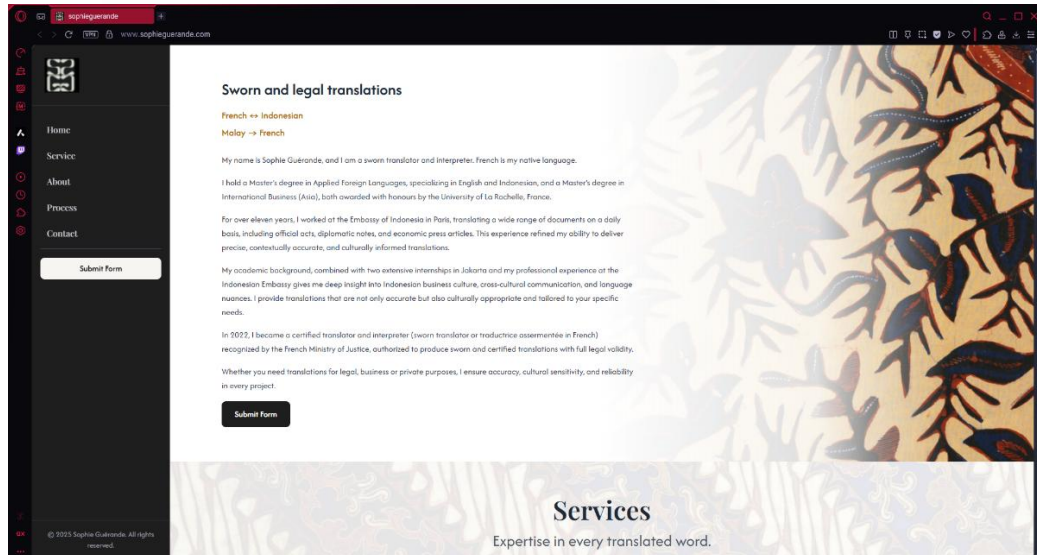
Seluruh konfigurasi sensitif seperti kredensial basis data PostgreSQL dan API key layanan email Brevo disimpan dalam environment variables yang diatur

langsung melalui dashboard Koyeb. Pendekatan ini memastikan keamanan data serta memisahkan konfigurasi dari kode sumber aplikasi. Laravel kemudian membaca variabel lingkungan tersebut melalui file `.env` pada saat aplikasi dijalankan.

Selain itu, konfigurasi penyimpanan file juga disesuaikan agar dokumen yang diunggah pengguna dapat tersimpan dengan aman dan dapat diakses oleh sistem ketika diperlukan untuk lampiran email. Setelah seluruh konfigurasi selesai, aplikasi backend dijalankan dan dilakukan pengujian ulang untuk memastikan endpoint API dapat diakses dengan baik dan email dapat terkirim melalui Brevo.

Dengan menggunakan Koyeb sebagai penyedia hosting backend, aplikasi dapat berjalan secara stabil dan mudah dikelola, serta mendukung kebutuhan sistem yang bersifat ringan namun membutuhkan keandalan tinggi. Deployment ini memungkinkan sistem backend untuk berfungsi sebagai penghubung utama antara frontend dan layanan email, sehingga proses komunikasi antara pengguna dan penerjemah dapat berjalan secara optimal.

Pengembangan Frontend Sistem:



Gambar 3.12 Hasil akhir website

Pengembangan frontend pada proyek ini bertujuan untuk menyediakan antarmuka yang sederhana namun fungsional, sehingga pengguna dapat dengan mudah mengirimkan permintaan layanan penerjemahan kepada penerjemah tersumpah. Frontend dibangun menggunakan React dengan Typescript, yang dipilih karena kemampuannya dalam mengelola komponen, state, serta komunikasi dengan backend melalui API.

Secara umum, frontend berperan sebagai pengumpul data dari pengguna dan pengirim data tersebut ke backend. Seluruh proses logika bisnis, validasi utama, serta pengiriman email tetap ditangani oleh backend, sehingga frontend dibuat sesederhana mungkin dan berfokus pada pengalaman pengguna.

Komponen utama pada frontend adalah formulir pengajuan (submission form). Formulir ini memungkinkan pengguna untuk mengisi data seperti nama, alamat email, nomor telepon, catatan tambahan, serta mengunggah dokumen yang akan diterjemahkan. Data tersebut dikelola menggunakan state React, sehingga setiap perubahan input dapat langsung disimpan dan dikontrol oleh sistem.

```

const handleSubmit = async (e: React.FormEvent) => {
  e.preventDefault();

  if (!formData.file) {
    setMessage("Please upload a file before submitting.");
    return;
  }

  const submission = new FormData();
  submission.append("name", formData.name);
  submission.append("email", formData.email);
  submission.append("phone_number", formData.phone_number);
  submission.append("notes", formData.notes);
  submission.append("file", formData.file);

  try {
    setLoading(true);
    setMessage(null);

    const res = await axios.post(
      "https://necessary-mora-jonathanryan2015-d56bceca.koyeb.app/api/submissions",
      // "http://127.0.0.1:8000/api/submissions",
      submission,
      // {
      //   headers: { "Content-Type": "multipart/form-data" },
      // }
    );

    setMessage("✅ Success: " + res.data.message);
    setFormData({
      name: "",
      email: "",
      phone_number: "",
      notes: "",
      file: null,
    });
  }
};

```

Gambar 3.13 Implementasi Fungsi Pengiriman Formulir pada Frontend React

Gambar 3.13 menampilkan implementasi fungsi `handleSubmit` pada frontend yang bertanggung jawab dalam menangani proses pengiriman formulir pengajuan terjemahan oleh pengguna. Fungsi ini dijalankan ketika pengguna menekan tombol submit pada form dan diawali dengan pemanggilan `preventDefault()` untuk mencegah perilaku bawaan browser agar halaman tidak melakukan reload.

Pada tahap awal, sistem frontend melakukan validasi dasar dengan memastikan bahwa pengguna telah mengunggah file dokumen sebelum formulir dikirimkan. Validasi ini bertujuan untuk mengurangi kemungkinan kesalahan pengiriman data ke backend serta meningkatkan pengalaman pengguna dengan memberikan umpan balik secara langsung apabila data belum lengkap.

Setelah validasi terpenuhi, seluruh data formulir, termasuk nama, alamat email, nomor telepon, catatan tambahan, dan file dokumen, dikemas menggunakan objek FormData. Penggunaan FormData diperlukan karena sistem harus mengirimkan data dalam format multipart/form-data, yang mendukung pengiriman file bersamaan dengan data teks dalam satu permintaan HTTP.

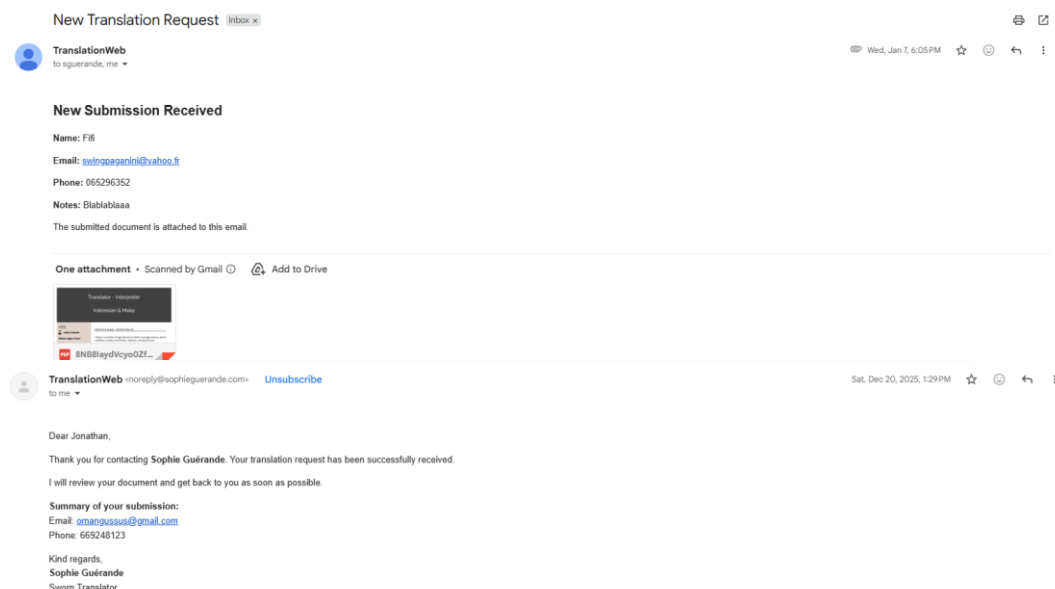
Proses pengiriman data ke backend dilakukan menggunakan library Axios melalui metode HTTP POST menuju endpoint API yang telah disediakan oleh backend. Selama proses pengiriman berlangsung, sistem mengaktifkan indikator loading untuk menandakan bahwa permintaan sedang diproses. Apabila pengiriman berhasil, frontend akan menampilkan pesan sukses berdasarkan respons dari backend dan secara otomatis mengosongkan kembali field formulir agar siap digunakan untuk pengajuan berikutnya.

Jika terjadi kesalahan selama proses pengiriman, sistem akan menangkap error tersebut dan menampilkan pesan kegagalan kepada pengguna. Pendekatan ini memastikan bahwa pengguna selalu mendapatkan informasi terkait status pengajuan yang dilakukan.

The screenshot displays the 'Submit a Document' form on the Sophie Guérande website. The form is part of a 4-step process: 1. Document Submission (Send document by email), 2. Quote & Payment (Receive quote and confirm), 3. Confirmation (Get order confirmation), and 4. Delivery (Receive translation by email). The form fields include: Full Name (text input), Email Address (text input), Phone Number (text input), and an Upload File button (with 'Choose File' and 'No file chosen' options). There is also a Notes section for additional details or instructions. A 'Submit' button is at the bottom of the form. The website footer includes the copyright notice '© 2016 Sophie Guérande. All rights reserved.', the name 'Sophie Guérande', and links for 'Services' and 'Legal Information'.

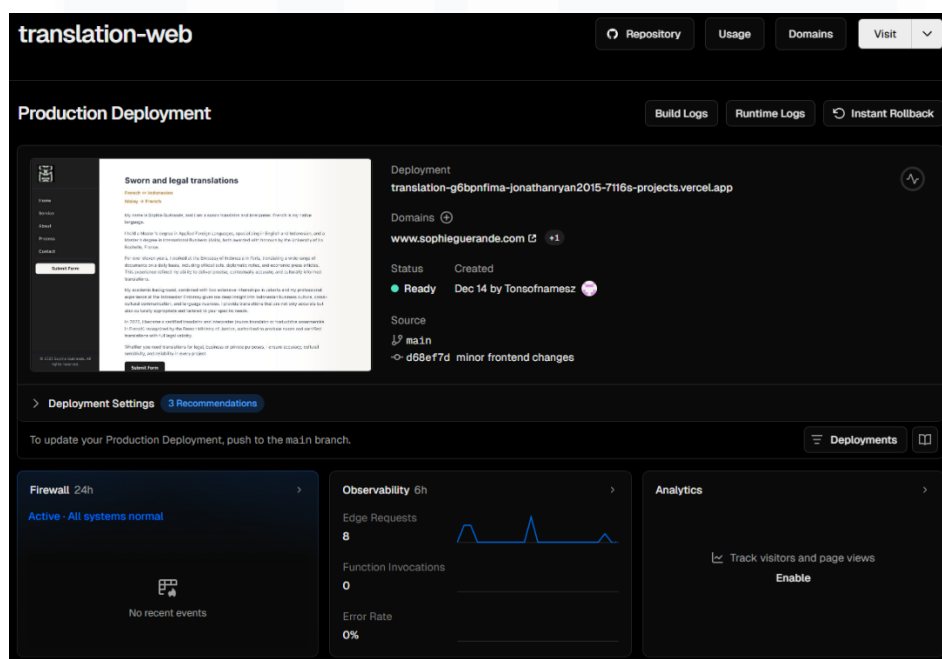
Gambar 3.14 Tampilan Formulir Pengajuan di Website

Gambar 3.14 menampilkan tampilan formulir pengajuan penerjemahan yang digunakan oleh pengguna untuk mengirimkan data identitas dan dokumen yang akan diterjemahkan. Melalui formulir ini, pengguna dapat mengunggah dokumen secara langsung serta menambahkan catatan tambahan sesuai kebutuhan. Setelah formulir dikirim dan data berhasil diproses oleh backend, sistem akan melanjutkan ke tahap pengiriman email otomatis.



Gambar 3.15 Tampilan Email Otomatis yang Diterima Penerjemah / Pengguna

Gambar 3.15 memperlihatkan contoh email notifikasi yang diterima oleh penerjemah tersumpah. Email ini berisi informasi pengajuan klien beserta dokumen yang dilampirkan secara otomatis oleh sistem. Selain itu, sistem juga mengirimkan email konfirmasi otomatis kepada pengguna sebagai tanda bahwa permintaan penerjemahan telah berhasil diterima. Dengan adanya fitur ini, proses komunikasi antara klien dan penerjemah menjadi lebih cepat, terstruktur, dan efisien tanpa memerlukan interaksi manual tambahan.



Gambar 3.16 Tampilan Dashboard Vercel untuk Monitoring Status Frontend

Proses deployment dilakukan dengan menghubungkan repository frontend ke Vercel. Setiap perubahan kode yang diunggah ke repository akan secara otomatis memicu proses build dan deployment oleh Vercel. Platform ini juga menyediakan konfigurasi environment variable apabila dibutuhkan, meskipun pada proyek ini frontend hanya memerlukan URL endpoint backend.

Vercel secara otomatis menyediakan sertifikat HTTPS serta optimasi performa melalui Content Delivery Network (CDN), sehingga aplikasi dapat diakses dengan cepat dan aman dari berbagai lokasi. Setelah proses deployment selesai, frontend

dapat diakses melalui domain yang telah ditentukan dan dihubungkan dengan domain utama yang dibeli melalui Hostinger.

3.3 Kendala yang Ditemukan

Selama pelaksanaan kerja magang dan pengembangan sistem website penerjemah tersumpah, terdapat beberapa kendala yang ditemui, meskipun secara umum proyek berjalan dengan lancar. Kendala yang muncul lebih banyak berkaitan dengan aspek non-teknis dibandingkan dengan kompleksitas sistem itu sendiri.

Kendala utama yang dihadapi adalah komunikasi dan koordinasi dengan supervisor. Supervisor merupakan pemilik usaha yang juga menjalankan seluruh operasional perusahaan secara mandiri. Kesibukan supervisor dalam menangani klien, pekerjaan penerjemahan resmi, serta tugas profesional lainnya menyebabkan komunikasi tidak selalu dapat dilakukan secara intensif dan real-time. Selain itu, seluruh proses koordinasi dilakukan secara daring karena perbedaan lokasi dan tidak adanya pertemuan tatap muka.

Kendala berikutnya adalah keterbatasan anggaran pengembangan sistem. Karena perusahaan berstatus auto-entrepreneur (wirausaha perorangan), anggaran untuk pengembangan website dan sistem backend sangat terbatas. Hal ini mempengaruhi pemilihan teknologi, layanan hosting, serta penyedia layanan pihak ketiga, sehingga diperlukan pertimbangan matang agar sistem tetap dapat berjalan optimal dengan biaya seminimal mungkin.

Selain itu, keterbatasan anggaran juga membatasi kemungkinan pengembangan fitur tambahan di luar kebutuhan utama sistem. Oleh karena itu, ruang lingkup proyek harus dijaga agar tetap fokus pada fungsi inti, yaitu pengiriman formulir permintaan penerjemahan melalui website ke email penerjemah.

3.4 Solusi atas Kendala yang Ditemukan

Untuk mengatasi kendala komunikasi dengan supervisor, solusi yang diterapkan adalah penjadwalan koordinasi secara fleksibel namun terstruktur. Komunikasi dilakukan secara daring melalui media pesan dan pertemuan online singkat, dengan

target minimal satu kali koordinasi dalam satu minggu apabila memungkinkan. Selain itu, setiap perubahan atau keputusan teknis yang telah disepakati selalu dirangkum secara tertulis untuk menghindari miskomunikasi dan mempercepat proses pengembangan.

Dalam menghadapi keterbatasan anggaran, solusi yang diambil adalah pemanfaatan layanan dan teknologi berbiaya rendah atau gratis tanpa mengorbankan stabilitas sistem. Backend dideploy menggunakan platform Koyeb, frontend menggunakan Vercel, serta layanan pengiriman email menggunakan Brevo yang menyediakan paket gratis dengan batasan tertentu. Pemilihan teknologi ini memungkinkan sistem tetap berjalan dengan baik, aman, dan dapat diakses secara publik tanpa biaya operasional yang besar.

Selain itu, ruang lingkup pengembangan sistem difokuskan pada kebutuhan utama perusahaan, yaitu penerimaan dan pengiriman permintaan layanan penerjemahan. Pendekatan ini membantu menghindari pengembangan fitur yang tidak esensial, sehingga waktu, biaya, dan sumber daya dapat dimanfaatkan secara lebih efektif.