SENATIK 2021, Vol. VII, P-ISSN :2337-3881, E-ISSN : 2528-1666 DOI: 10.28989/senatik.v7i1.449

Comparison of Eloquent ORM with Query Builder in Work Management System (Case Study: Muhammadiyah Lamongan Hospital)

Febryan Akhdani [1*] and Danur Wijayanto[1]
Information Technology, Faculty of Science and Technology, Universitas 'Aisyiyah Yogyakarta, Indonesia

*Email corespondensi : akhdanifebryan@gmail.com
Received Feb 03, 2022; Accepted Feb 20, 2022; Published Mar 08, 2022

Abstract. A management system is a set of policies, processes, and procedures used by an organization to ensure that the system can fulfill the tasks required to achieve its objectives. With this system, the company can make it easier for the management of a task/project that is being done by the employee/employee. In the SIRS Unit at Lamongan Muhammadiyah Hospital, the implementation of Work Management is still done manually, namely by holding monthly meetings held at the beginning of the month which can affect the efficiency and effectiveness of employees in terms of collecting/reporting tasks assigned by the project leader. For this reason, a Job Management System is needed that can overcome these problems. Job Management System, created on a Web-based basis using Laravel 8.0 as the framework and MySQL as the database. In this study, researchers will focus on implementing Eloquent ORM which is a special feature in Laravel to make it easier for developers to process data and compare Eloquent ORM with Query builder in its implementation on Job Management System. The result of this research is Query Builder can execute database query more faster and more memory efficient than Eloquent, but Eloquent use simple code than Eloquent.
Keywords: *Laravel, Eloquent, Query Builder, Database, MySQL*

1. Pendahuluan

Rumah Sakit Muhammadiyah Lamongan merupakan lembaga Amal Usaha Muhammadiyah(AUM) yang bergerak pada bidang kesehatan yang terletak di Jl. Jaksa Agung Suprapto No.76, Sarirejo, Sukorejo, Kec. Lamongan, Kabupaten Lamongan, Jawa Timur. Saat ini Rumah Sakit Muhammadiyah Lamongan telah berkembang menjadi Rumah Sakit Tipe B dan telah memenuhi Standar Akreditasi dengan predikat PARIPURNA. Pencapaian Rumah Sakit menjadi Tipe B diperoleh pada 23 Oktober 2013, dan penghargaan Akreditasi RS diperoleh pada tahun 2014 dan tahun 2017 dengan predikat Lulus Paripurna. Penghargaan Akreditasi RS ini merupakan wujud dari upaya Rumah Sakit dalam menjaga mutu dan keselamatan pasien. Instalasi Gizi Rumah Sakit Muhammadiyah Lamongan juga memperoleh sertifikat Halal dari Majelis Ulama Indonesia (MUI). Pemenuhan halal terhadap produk makanan instalasi gizi merupakan wujud komitmen RS sebagai Rumah Sakit Syariah [1]. Terdapat banyak unit di RSML guna meningkatkan kualitas pelayanan dan mutu rumah sakit, seperti Unit Gizi, Unit Depo Farmasi, Unit SIRS, dll.

Unit SIRS merupakan unit yang bergerak dalam bidang sistem informasi rumah sakit yang bertujuan untuk meningkatkan kualitas sistem informasi RSML [2]. SIRS juga bertanggung jawab terhadap semua sistem informasi yang ada pada RSML, seperti sistem infromasi pendaftaran, sistem informasi poli, sistem informasi depo farmasi, dll. Namun, pada Unit SIRS terdapat satu problematika yaitu belum adanya sebuah sistem untuk memanajemen pekerjaan dari karyawan yang ada Unit SIRS, mengingat tanggung jawab Unit SIRS mencakup seluruh program atau aplikasi yang ada pada RSML. Untuk itu perlu dibuatkanlah sebuah Sistem Manajemen Pekerjaan yang dapat membantu karyawan Unit SIRS dalam memanajemen pekerjaan yang diberikan. Sistem Manajemen Pekerjaan juga membantu kepala bagian SIRS untuk memonitoring progress pekerjaan yang sedang dikerjakan oleh karyawan Unit SIRS sehingga pekerjaan yang sedang dikerjakan oleh karyawan Unit SIRS bisa termanajemen dengan baik.
Dalam perancangan Sistem Manajemen Pekerjaan ini, sistem akan dibuat dengan berbasis *website* dengan *Framework Laravel* dan *Database MySQL*. *Laravel* adalah framework aplikasi *web* dengan sintaks yang ekspresif dan elegan. Kerangka kerja *web* menyediakan struktur dan titik awal untuk membuat aplikasi, memungkinkan *user* untuk fokus menciptakan sesuatu yang luar biasa sementara developer membahas detailnya [3]. Pada *Laravel* terdapat sebuah fitur khusus yaitu *Eloquent ORM*. *Eloquent ORM* adalah sebuah fitur yang terdapat pada *Laravel*, dimana fitur ini dapat membantu developer dalam memanipulasi dan mengakses *database* dengan perintah yang lebih singkat dan mempercepat proses pembuatan CRUD dari *database*. Salah satu hal yang membuat *Eloquent ORM* menjadi menarik adalah penggunaan *query* pada *MySQL*, sebagai contoh jika pada PHP Native menggunakan *query* SELECT * FROM (nama_tabel) untuk menampilkan semua isi dari sebuah table, maka pada *Eloquent ORM* hanya perlu menggunakan all() untuk menampilkan semua isi dari sebuah table. Selain menggunakan *Eloquent ORM* untuk memanipulasi tabel pada *Laravel*, ada satu fitur lagi untuk memanipulasi tabel pada *Laravel* yaitu *Query builder*. Mirip dengan *Eloquent ORM*, *Query builder* merupakan sebuah fitur yang ada pada *Laravel* yang digunakan untuk membuat dan menjalankan *query* pada *database*. *Query builder* menggunakan *(PHP Data Object)* PDO *parameter binding* untuk melindungi aplikasi kita dari serangan injeksi SQL, sehingga *developer* tidak perlu melakukan filtering.
Eloquent ORM dan *Query Builder* sering digunakan oleh developer Laravel dalam membangun sebuah website, baik itu website blog maupun sebuah sistem yang menggunakan basis website. sehingga penelitian ini berfokus pada perbandingan performa *Eloquent ORM* dengan *Query builder*, sehingga dapat membantu developer dalam mentukan fitur mana yang nantinya bisa digunakan dalam mengembangkan aplikasi. Adapun batasan-batasan masalah yang ada pada penelitian ini antara lain, perancangan sebuah Sistem Manajemen Pekerjaan dengan studi kasus Unit SIRS pada Rumah Sakit Muhammadiyah Lamongan dengan fokus pada pengelolaan data dari *database* menggunakan fungsi *Eloquent ORM* dan *Query builder*, tidak membahas terkait keamanan *database* yang dibuat. Selain itu, alasan penulis memilih untuk membandingkan fungsi

2. Metodologi Penelitian

Dalam subbab ini, akan dijelaskan mengenai metode yang digunakan dalam penelitian ini dalam membandingkan fungsi Eloquent ORM dan Query builder pada framework Laravel. Terdapat 3 tahapan yang penulis lakukan dalam penelitian ini, yang pertama adalah mengumpulkan data-data yang diperlukan dengan metode wawancara dan studi literatur, melakukan perancangan dan pembuatan Sistem Manajemen Pekerjaan, dan yang terakhir adalah melakukan perbandingan Eloquent ORM dan Query builder dengan beberapa metode perbandingan.

2.1. Wawancara

Wawancara merupakan salah satu teknik yang dapat digunakan untuk mengumpulkan data penelitian. Secara sederhana dapat dikatakan bahwa wawancara (interview) adalah suatu kejadian atau suatu proses interaksi antara pewawancara (interviewer) dan sumber informasi atau orang yang diwawancarai (interviewee) melalui komunikasi langsung. Wawancara bertujuan mencatat opini, perasaan, emosi, dan hal lain berkaitan dengan individu yang ada dalam organisasi. Dengan melakukan interview, peneliti dapat memperoleh data yang lebih banyak sehingga peneliti dapat memahami budaya melalui bahasa dan ekspresipi hak dari interviewee; dan dapat melakukan klarifikasi atas hal- hal yang tidak diketahui [4].

Dalam penelitian ini, wawancara dilakukan untuk mendapatkan data primer terkait Unit SIRS yang ada pada Rumah Sakit Muhammadiyah Lamongan berdasarkan persepsi kepala bagian dan karyawan senior yang ada pada Unit SIRS. Selain itu, wawancara juga dilakukan untuk mendapatkan informasi terkait requirement system yang diinginkan oleh Unit SIRS dalam aplikasi sistem yang sedang dikerjakan.

2.2. Studi Literatur

Studi Literature atau studi kepustakaan adalah teknik pengumpulan data dengan mengadakan studi penelaahan terhadap buku-buku, literatur-literatur, catatan-catatan, dan laporan-laporan yang ada hubungannya dengan masalah yang dipecahkan [5]. Menurut Nazir [5] studi kepustakaan merupakan langkah yang penting dimana setelah seorang peneliti menetapkan topik penelitian, langkah selanjutnya adalah melakukan kajian yang berkaitan dengan teori yang berkaitan dengan topik penelitian. Dalam pencarian teori, peneliti akan mengumpulkan informasi sebanyak-banyaknya dari kepustakaan yang berhubungan. Sumber-sumber kepustakaan dapat diperoleh dari: buku, jurnal, majalah, hasil-hasil penelitian (tesis dan disertasi), dan sumber-sumber lainnya yang sesuai (internet, koran dll).
Sedangkan menurut Daniah dan Warsiah [6], Studi Literatur adalah merupakan penelitian yang dilakukan oleh peneliti dengan mengumpulkan sejumlah buku buku, majalah yang berkaitan dengan masalah dan tujuan penelitian. Dalam penelitian ini, studi literatur dilakukan untuk mendapatkan data informasi sebanyak mungkin untuk mendukung hasil dari penelitian ini. Studi literatur yang digunakan berupa jurnal-jurnal perancangan sistem yang menggunakan Laravel dan MySQL, dokumentasi dari framework yang digunakan, serta beberapa artikel yang membahas terkait pembahasan seputar penelitian ini.
Laravel adalah framework aplikasi web dengan sintaks yang ekspresif dan elegan. Kerangka kerja web menyediakan struktur dan titik awal untuk membuat aplikasi, memungkinkan user untuk fokus menciptakan sesuatu yang luar biasa sementara developer membahas detailnya [3]. Framework Laravel pertama kali dibuat oleh Taylor Otwell sebagai alternatif framework dari bahasa pemrograman PHP yang memudahkan developer web untuk membuat rangka bingkai kerja sebuah website. Laravel pertama kali diperkenalkan pada 09 Juni 2011 dengan versi beta. Dilansir dari situs Techgig, bahwasannya Laravel merupakan framework PHP terpopuler dan terbaik hingga saat ini, dan paling banyak digemari oleh para developer website dalam membangun sebuah website [7].
MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS yang multialur, multipengguna, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus di mana penggunaannya tidak cocok dengan penggunaan GPL. MySQL pada awalnya diciptakan pada tahun 1979, oleh Michael "Monty" Widenius, seorang programmer komputer asal Swedia. Monty mengembangkan sebuah sistem database sederhana yang dinamakan UNIREG yang menggunakan koneksi low-level ISAM database engine dengan indexing [8].
Eloquent ORM adalah teknik untuk mengubah data dari objek ke dalam sebuah relasi, atau tabel (jika dalam pengertian DBMS). Kehadiran *Eloquent ORM* menghilangkan kebutuhan pembacaan data reasional dari DBMS (katakanlah *MySQL*) yang sangat kompleks ke dalam objek. Selain itu, *Eloquent ORM* juga dapat meringkas beberapa fungsi pemetaan logika seperti penggunaan *create and read* dan penggunaan *relationship* tabel pada sebuah *database* [9]. Penggunaan kode *Eloquent ORM* pada *Laravel* ditunnjukkan pada Gambar 1.

```
{
    //
    public function index($id)
    {
        $prodet = Proyek::find('id');
        return view('proyek.projects_detail', ['proyeks' => $prodet]);
    }
}
```

Gambar 1. Kode Eloquent ORM

Sedangkan *Query builder*, merupakan sebuah fungsi yang disediakan oleh SQL yang dirancang untuk meningkatkan produktivitas dan menyederhanakan tugas pembuatan kueri SQL. Didalam framework

Laravel, *Query builder* hadir dengan dukungan PDO parameter yang mana berfungsi sebagai pengikat parameter untuk melindungi aplikasi yang sedang kita bangun dari serangan injeksi SQL. Dengan adanya PDO parameter pada *Query builder*, menjadikan PDO parameter sebagai salah satu kelebihan dari penggunaan *Query builder* pada sebuah *website* agar tidak terjadi kebocoran pada SQL. Penggunaan *syntax Query builder*, adalah seperti kode berikut

```
        {
            $proyeks = DB::table('proyeks')->get();

            return view('proyeks.projects', ['proyeks' => $proyeks]);
        }
    }
```

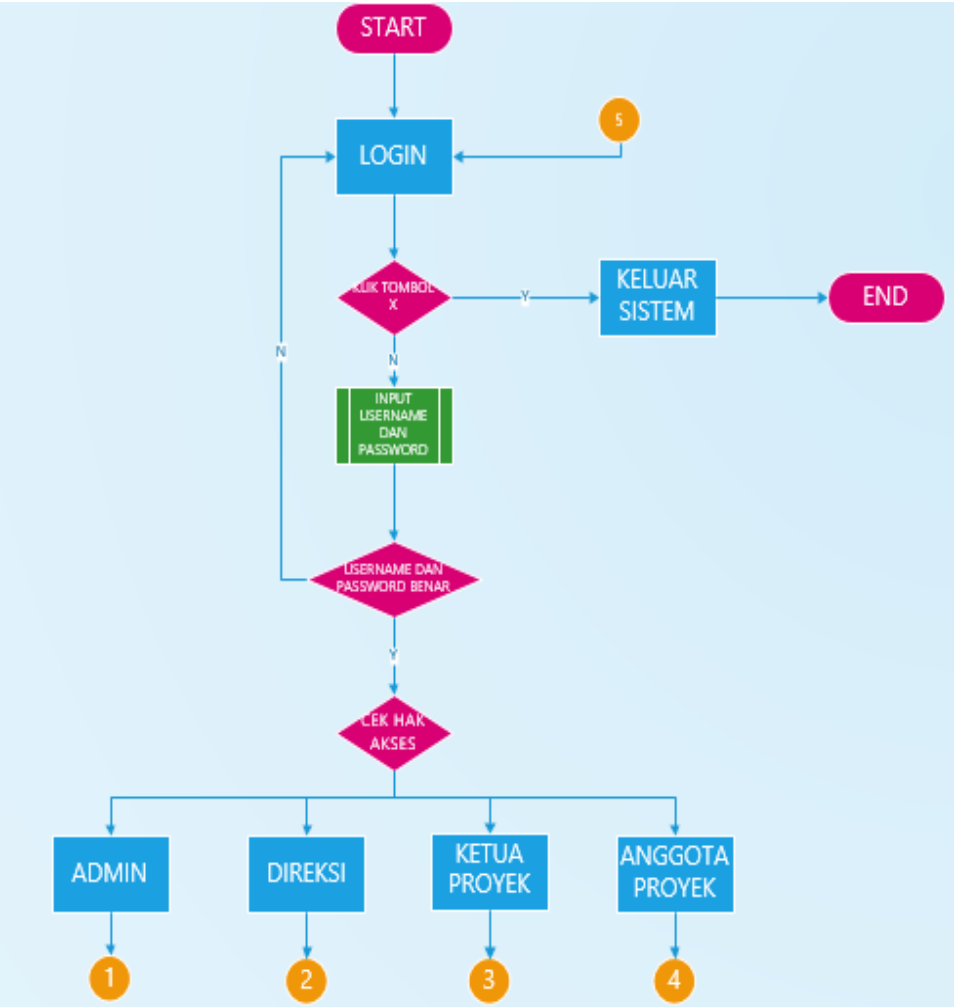
Gambar 2. Kode Query Builder

Pada *syntax* di Gambar 2, fungsi dari *Query builder* terletak pada baris ke 3, yakni pada `$proyeks = DB::table('proyeks')->get();`adalah untuk menampilkan semua isi dari tabel proyeks kedalam variable `$proyeks`. Lalu untuk fungsi `return view`, jika *user* menggunakan atau mengakses fungsi `index()`, maka *user* akan diarahkan ke halaman `proyeks.projects` serta melakukan data passing yakni variabel `$proyeks`.

2.3. Perancangan Sistem Manajemen Pekerjaan

Dalam penelitian ini, studi kasus yang diambil adalah Sistem Manajemen Pekerjaan yang sedang dirancang dan dibuat oleh penulis bersama tim yang membuat. Pembuatan sistem manajemen ini menggunakan framework Laravel sebagai bingkai kerja PHP nya dan MySQL sebagai DBMS (Database Management System) untuk mengelola dan memanipulasi data pada sistem. Dalam perancangan dan pembuatan Sistem Manajemen Pekerjaan, penulis menggunakan metode Waterfall karena metode tersebut merupakan metode yang paling cocok digunakan pada rancang bangun Sistem Manajemen Pekerjaan.

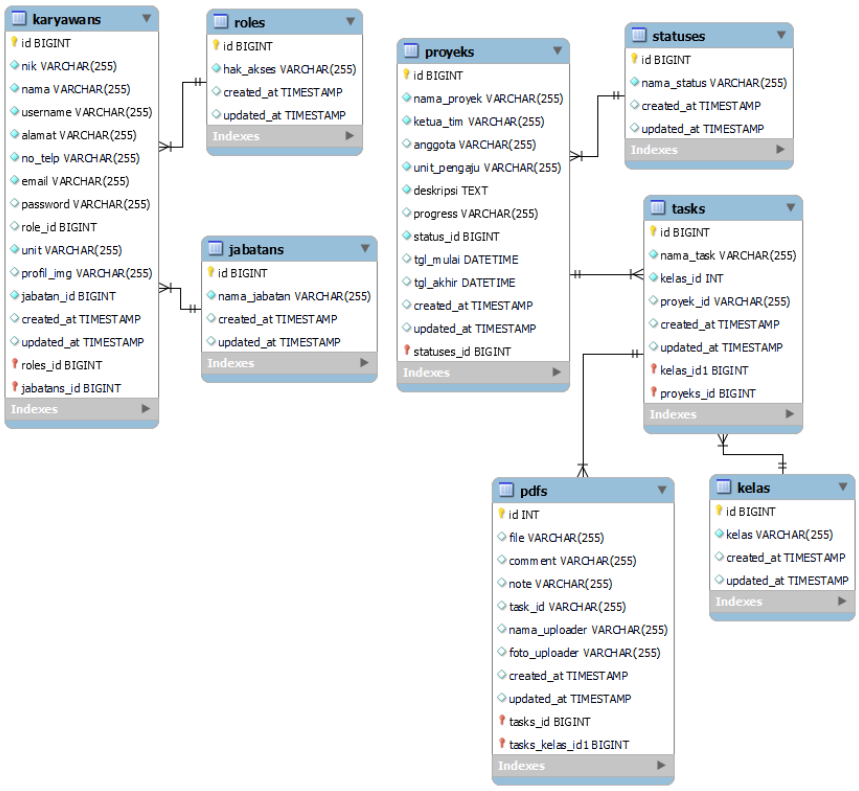
Untuk tahapan pertama yaitu perancangan diagram alur dari Sistem Manajemen Pekerjaan. Diagram alur atau flowchart adalah sebuah representasi grafis dari urutan operasi dalam sistem informasi atau program. Bagan alir sistem informasi menunjukkan bagaimana data mengalir dari dokumen sumber melalui komputer ke distribusi akhir kepada pengguna. Bagan alir program menunjukkan urutan instruksi dalam satu program atau subrutin. Simbol yang berbeda digunakan untuk menggambar setiap jenis diagram alur [9]. Gambar 3 menunjukkan *flowchart* dari Sistem Manajemen Pekerjaan



Gambar 3. Flowchart

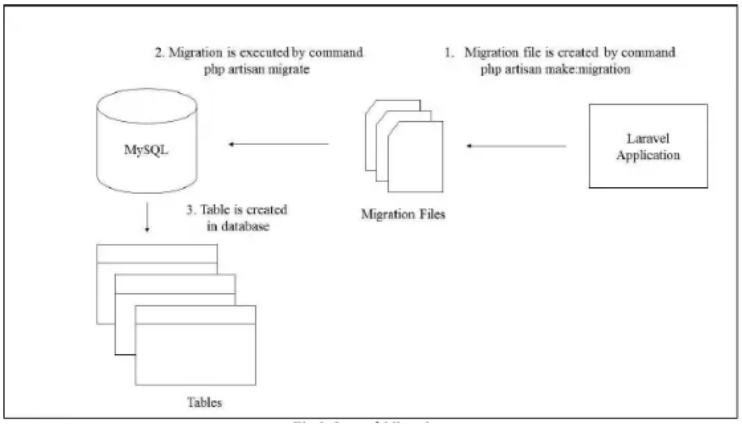
Selanjutnya adalah perancangan diagram *use case*. Diagram use case digunakan untuk mengumpulkan persyaratan sistem termasuk pengaruh internal dan eksternal. Persyaratan ini sebagian besar persyaratan desain. Jadi ketika suatu sistem dianalisis untuk mengumpulkan fungsionalitasnya, use case disiapkan dan aktor diidentifikasi [11]. Pada Sistem Manajemen Pekerjaan, terdapat 4 *role user*, yang mana keempat role ini dapat membatasi penggunaan sistem tergantung dari role yang sedang login. Misalnya pada *role* Admin, bisa mengakses semua menu yang ada pada Sistem Manajemen Pekerjaan ini. Sedangkan pada user Anggota Proyek, hanya bisa menggunakan menu Task, Kalender, Chat, dan Menu Home Anggota Proyek.

Selanjutnya adalah proses migration database pada Laravel supaya data yang ada pada database bisa dimanipulasi melalui kode Laravel. Selain itu, penulis bersama tim juga membuat diagram ERD (Entity Relationship Diagram) yang mana diagram ini dapat membantu pembaca agar lebih mudah dalam penggunaan data pada database yang ditunjukkan pada Gambar 4.



Gambar 3. Diagram ERD

Pada Gambar 4, tabel karyawan memiliki relasi dengan tabel roles dan jabatans yaitu 1 : N untuk masing-masing tabel, dimana pada column role_id dan jabatan_id merupakan foreign key dari column id pada tabel roles dan id dari tabel jabatans. Sedangkan pada tabel proyeks, memiliki relasi terhadap tabel statuses dengan relasi 1 : N, dimana column status_id merupakan foreign key dari column id pada tabel statuses. Pada tabel tasks, memiliki relasi dengan tabel proyeks, kelas, dan pdfs. Untuk tabel proyeks dan pdfs memiliki relasi N : 1 dengan proyeks_id dan kelas_id sebagai foreign key-nya, sedangkan untuk tabel pdfs, memiliki relasi 1 : N dengan column task_id sebagai foreign key-nya.
Tabel karyawan memiliki fungsi untuk menyimpan data karyawan yang terhubung dengan tabel roles dan jabatans. Tabel roles memiliki fungsi sebagai tabel master dari hak akses yang dimiliki oleh karyawan, contoh Admin. Sedangkan tabel jabatans, memiliki fungsi sebagai tabel master dari jabatan yang dimiliki oleh setiap karyawan, contoh Karyawan. Tabel proyeks, memiliki fungsi sebagai tabel yang menyimpan data proyek yang telah ditambahkan oleh user. Pada tabel proyeks, memiliki relasi dengan tabel status, yang mana tabel status merupakan tabel master dari status yang dimiliki oleh setiap proyek yang ada, contoh On Progress. Tabel task memiliki fungsi sebagai tempat meyimpan data task yang dibuat oleh user. Pada tabel task, memiliki relasi dengan tabel kelas proyek, pdfs. Tabel kelas berfungsi sebagai tabel master dari kelas yang dimiliki oleh setiap task, seperti tugas, progress dan selesai. Dan untuk proyeks, memiliki fungsi sebagai identitas bahwa task berada pada proyek tertentu, sehingga 1 task hanya bisa diakses dan dilihat dari satu proyek. Dan untuk pdfs, memiliki fungsi sebagai tabel transaksi dari file dan komentar yang diupload oleh user, contoh file ‘contoh.pdf’ dengan kometar ‘untuk task 1’.



Gambar 4. Laravel Migration Schema

Pada Gambar 3, tabel karyawan memiliki relasi dengan tabel roles dan jabatans yaitu 1 : N untuk masing-masing tabel, dimana pada column role_id dan jabatan_id merupakan foreign key dari column id pada tabel roles dan id dari tabel jabatans. Sedangkan pada tabel proyeks, memiliki relasi terhadap tabel statuses dengan relasi 1 : N, dimana column status_id merupakan foreign key dari column id pada tabel statuses. Pada tabel tasks, memiliki relasi dengan tabel proyeks, kelas, dan pdfs. Untuk tabel proyeks dan pdfs memiliki relasi N : 1 dengan proyeks_id dan kelas_id sebagai foreign key-nya, sedangkan untuk tabel pdfs, memiliki relasi 1 : N dengan column task_id sebagai foreign key-nya.

Tabel karyawan memiliki fungsi untuk menyimpan data karyawan yang terhubung dengan tabel roles dan jabatans. Tabel roles memiliki fungsi sebagai tabel master dari hak akses yang dimiliki oleh karyawan, contoh Admin. Sedangkan tabel jabatans, memiliki fungsi sebagai tabel master dari jabatan yang dimiliki oleh setiap karyawan, contoh Karyawan. Tabel proyeks, memiliki fungsi sebagai tabel yang menyimpan data proyek yang telah ditambahkan oleh user. Pada tabel proyeks, memiliki relasi dengan tabel status, yang mana tabel status merupakan tabel master dari status yang dimiliki oleh setiap proyek yang ada, contoh On Progress. Tabel task memiliki fungsi sebagai tempat meyimpan data task yang dibuat oleh user. Pada tabel task, memiliki relasi dengan tabel kelas proyek, pdfs. Tabel kelas berfungsi sebagai tabel master dari kelas yang dimiliki oleh setiap task, seperti tugas, progress dan selesai. Dan untuk proyeks, memiliki fungsi sebagai identitas bahwa task berada pada proyek tertentu, sehingga 1 task hanya bisa diakses dan dilihat dari satu proyek. Dan untuk pdfs, memiliki fungsi sebagai tabel transaksi dari file dan komentar yang diupload oleh user, contoh file ‘contoh.pdf’ dengan kometar ‘untuk task 1’.

Pada Gambar 4, menjelaskan alur dari proses migration database pada Laravel. Untuk tahap awal migration, user akan melakukan command php artisan make:migration lalu akan dibuatkan migration files. Selanjutnya di MySQL, data migrasi akan dieksekusi lalu MySQL akan membuatkan tabel baru di database yang dituju.

Bagian terakhir adalah diagram DFD. Diagram DFD (Data Flow Diagram) merupakan diagram yang menunjukkan alur penyebaran data yang terjadi pada sebuah sistem yang sedang berjalan. Dalam pembuatannya, diagram DFD memiliki 3 level, yang pertama adalah diagram konteks (level 0), DFD level 1, dan DFD level 2.

2.4. Pengujian Perbandingan

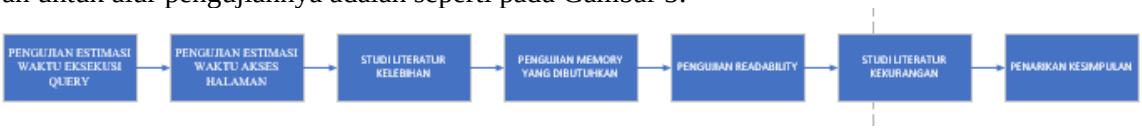
Untuk melakukan pengujian, penulis telah membuat beberapa aspek yang akan diuji, yaitu seperti yang ditunjukkan pada Tabel 1. Pengujian yang berada pada nomor 1, 2, 3, dan 4 akan diuji sebanyak 10 kali sehingga diharapkan mendapatkan hasil yang akurat.

Dalam pengujian estimasi waktu, dilakukan eksekusi sebuah query yang menampilkan data seluruh karyawan sebanyak 107 datadari database. Data tersebut mempunyai relasi dengan tabel roles yang memiliki 5 data dan tabel jabatan yang memiliki 4 data. Lalu data yang tampil adalah nama karyawan, jabatannya, dan hak aksesnya. Untuk melihat estimasi waktu yang diperlukan untuk mengeksekusi sebuah query, dapat dilihat pada tools Laravel Debugbar. Sedangkan dalam pengujian estimasi waktu akses halaman, peneliti menguji dari waktu refresh sampai tampilan website benar - benar dapat dilihat dan dapat difungsikan oleh pengguna.

Tabel 1. Tabel Indikator Pengujian

No	Indikator
1	Estimasi waktu eksekusi *query*
2	Estimasi waktu akses halaman
3	Memory yang dibutuhkan
4	*Readability*

Dan untuk alur pengujiannya adalah seperti pada Gambar 5.



Gambar 5. Alur Pengujian

Tahap pertama dari pengujian adalah pengujian terkait estimasi waktu eksekusi query dari kedua subjek penelitian yaitu *Eloquent* dan *Query Builder*. Tujuan dari pengujian ini adalah untuk mengetahui fungsi mana yang lebih cepat dari aspek estimasi waktu eksekusi query ketika sebuah halaman pada website diakses dan menampilkan data yang diambil dari database. Tahap selanjutnya yakni pengujian estimasi waktu akses halaman. Sedikit sama dengan tahap sebelumnya, namun pada tahap ini bertujuan untuk mengetahui seberapa cepat sebuah halaman bisa diakses oleh user, yang mana halaman tersebut akan menampilkan data yang diambil dari database.
Tahap pengujian yang ketiga adalah studi literatur kelebihan dari kedua subjek penelitian, dilanjutkan dengan studi literatur kekurangan dari kedua subjek penelitian yang mana tahap ini nanti akan akan digabung menjadi satu di akhir sebelum penarikan kesimpulan. Jadi setelah tahap pengujian estimasi waktu akses halaman, tahap selanjutnya adalah pengujian memory yang dibutuhkan dalam mengakses halaman website yang menampilkan data dari database. Dan pengujian selanjutnya adalah pengujian *readability syntax* atau kemudahan dalam membaca kode. Tujuan dari tahap pengujian ini adalah untuk mengetahui kode mana dari fungsi Eloquent dan Query Builder yang lebih mudah dibaca oleh developer Laravel yang masih awam maupun yang sudah ahli.
Dan tahap akhir dari pengujian ini adalah penarikan kesimpulan berdasarkan tahapan-tahapan pengujian yang sudah dilakukan sebelumnya. Hasil dari penarikan kesimpulan akan dijelaskan pada bab ke-4 yakni bab kesimpulan

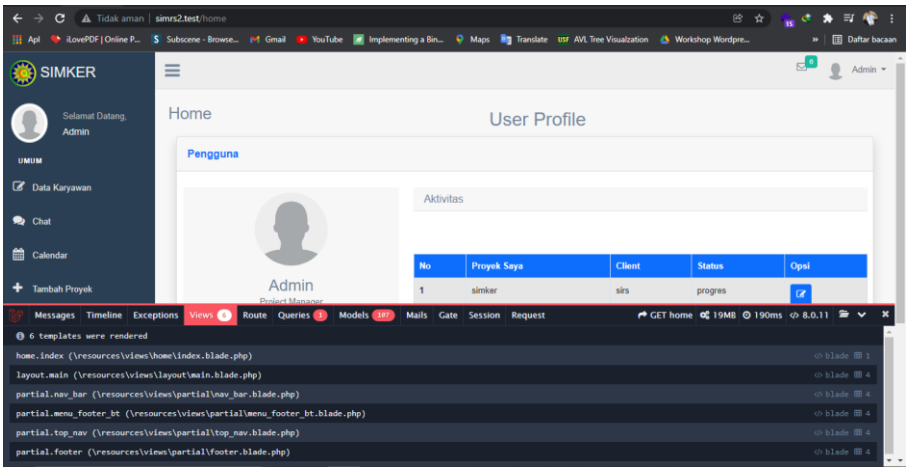
3. Hasil dan Pembahasan

Pada bab ini, penulis akan menyajikan hasil dan tahap pengujian. Pengujian memiliki tujuan untuk mengetahui manakah yang lebih baik antara halamn website yang menggunakan *Query Builder* dengan halaman yang menggunakan *Eloquent*. Pengujian yang dilakukan adalah seperti berikut: pengujian estimasi waktu eksekusi *query*, pengujian eksekusi waktu akses halaman, pengujian memory yang dibutuhkan ketika mengakses halaman, *comparing Readability Syntax* dari kedua objek penelitian, dan yang terakhir adalah *comparing* kelebihan dan kekurangan dari kedua objek penelitian.

3.1. Hasil

Untuk melakukan perbandingan antara Eloqeunt dengan *Query builder*, penulis menggunakan beberapa metode pengujian dan studi literatur terhadap beberapa sumber yang dapat dipercaya. Sedangkan untuk indikator pengujiannya adalah seperti pada Tabel 1. Tahap pertama adalah pengujian dengan indikator pada Table 1. Untuk melakukan uji estimasi waktu yang diperlukan untuk mengeksekusi sebuah *query*, diperlukan sebuah *tools* pihak ketiga yaitu *Laravel Debugbar* yang dibuat oleh Barry vd. Heuvel di situs GitHub. Untuk menginstall *Laravel Debugbar*, yang pertama jalankan perintah `composer require barryvdh/laravel-debugbar`, lalu tunggu hingga prosesnya selesai

Jika sudah selesai, buka file pada direktori config/app.php lalu tambah kode berikut pada bagian Provides `Barryvdh\Debugbar\ServiceProvider::class,` dan pada bagian Aliases `'Debugbar'=>Barryvdh \Debugbar\Facade::class,`
Langkah terakhir dalam install *Laravel Debugbar* adalah melakukan publish config dengan cara `php artisan vendor:publish --provider="Barryvdh\Debugbar\ServiceProvider"`, lalu tunggu hingga proses *publish* selesai. Jika sudah selesai, maka ketika membuka *website* atau projek yang sedang kita bangun di browser, muncul sebuah *tools* seperti Gambar 6.



Gambar 6. Tampilan Laravel Debugbar

Tabel 2 menunjukkan hasil dari tahap pengujian yang menunjukkan *Query Builder* lebih unggul dibandingkan dengan *Eloquent* dari segi estimasi waktu eksekusi *query*, waktu akses halaman, *memory*. Untuk pembahasan yang lebih detail akan dijelaskan pada sub bab selanjutnya.

Tabel 2. Hasil Dari Pengujian

No	Indikator	*Eloquent*	*Query builder*
1	Estimasi waktu eksekusi *query*		v
2	Estimasi waktu akses halaman		v
3	Memory yang dibutuhkan		v
4	*Readability syntax*	v	

3.2. Pembahasan

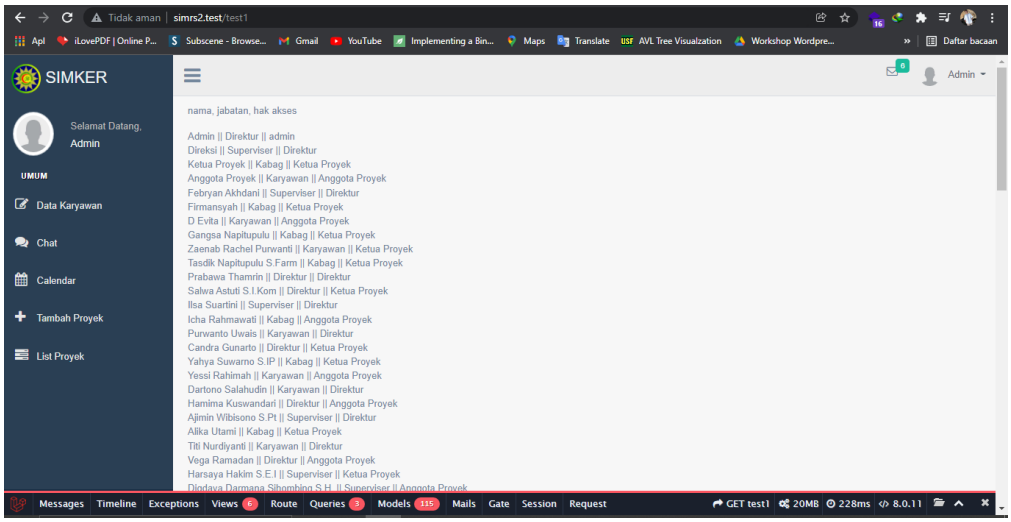
Dalam pengujian estimasi waktu untuk mengeksekusi sebuah *query* yang menampilkan data dari *database*, penulis akan menggunakan halaman baru yaitu simrs2.test/test1 untuk halaman yang menggunakan *Eloquent* dan simrs2.test/test2 untuk halaman yang menggunakan *Query builder*. Dan untuk data yang ditampilkan adalah seluruh data karyawan sebanyak 107 data yang terelasi dengan tabel *roles* yang memiliki 5 data dan tabel jabatan yang memiliki 4 data. Lalu data yang tampil adalah nama karyawan, jabatannya, dan hak aksesnya. Gambar 7 menunjukkan *syntax* untuk test 1 dan test 2 yang ada pada *controller*.

```
    public function test1(){
        $karyawans = Karyawan::with('role', 'jabatan')->get();
        return view('test', compact('karyawans'));
```

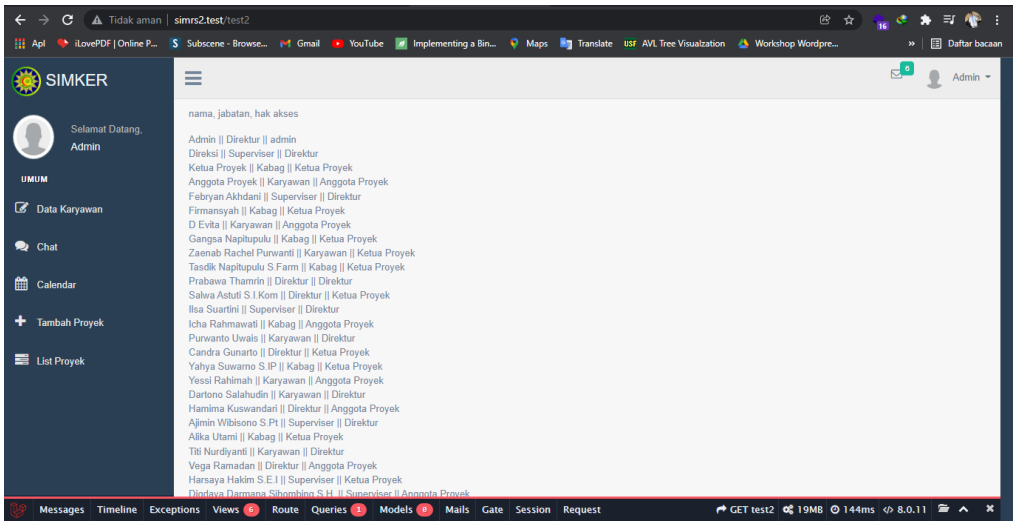
```
    }

    public function test2(){
        $karyawans = DB::table('karyawans')
        ->join('roles', 'karyawans.role_id', '=', 'roles.id')
        ->join('jabatans', 'karyawans.jabatan_id', '=', 'jabatans.id')
        ->select('nama', 'hak_akses', 'nama_jabatan')->get();
        return view('test2', compact('karyawans'));
    }
```

Gambar 7. Kode Query Builder

Dan untuk tampilan view dari test1, adalah seperti pada Gambar 8 dan untuk tampilan view dari test2 adalah seperti pada Gambar 9.



Gambar 8. Tampilan View Test1



Gambar 9. Tampilan View Test2

Lalu data yang tampil adalah nama karyawan, jabatannya, dan hak aksesnya. Dan untuk melihat estimasi waktu yang diperlukan untuk mengeksekusi sebuah *query*, dapat dilihat pada *tools Laravel Debugbar*. Untuk menghindari pengambilan keputusan yang kurang konkrit, penulis akan mengambil sebanyak 10 kali sampel yaitu dengan cara me-*refresh* halaman sebanyak 10 kali, dan hasil dari *refresh* tersebut akan

digunakan sebagai data pengujian. Tabel 3 menunjukkan hasil data pengujian estimasi waktu eksekusi *query*.

Tabel 3. Data Pengujian Estimasi Waktu Eksekusi *Query*

No	*Eloquent*	*Query builder*
1	24.36 ms	16.98 ms
2	16.74 ms	14.61 ms
3	28.92 ms	23.94 ms
4	19.79 ms	18.97 ms
5	16.34 ms	14.02 ms
6	14.34 ms	19.00 ms
7	24.56 ms	15.77 ms
8	26.56 ms	15.19 ms
9	24.46 ms	18.87 ms
10	29.77 ms	3.26 ms

Dari data yang ada pada Tabel 3, dapat dilihat bahwa rata-rata kecepatan eksekusi *Query Builder* lebih cepat daripada kecepatan eksekusi *Eloquent*. Percobaan ke 9 dari 10 sampel menunjukkan bahwa *Query Builder* lebih unggul dari *Eloquent* dalam segi estimasi waktu eksekusi sebuah *query*.
Pengujian selanjutnya adalah estimasi waktu akses sebuah halaman. Tabel 4 menunjukkan pengujian dari hasil *refresh* untuk menguji estimasi waktu akses sebuah halaman.

Tabel 4. Data Sampel Pengujian Estimasi Waktu Akses Halaman

No	*Eloquent*	*Query builder*
1	204 ms	176 ms
2	216 ms	156 ms
3	201 ms	168 ms
4	203 ms	159 ms
5	194 ms	156 ms
6	163 ms	159 ms
7	171 ms	157 ms
8	167 ms	171 ms
9	184 ms	194 ms
10	165 ms	156 ms

Dari hasil pengujian yang ditunjukkan pada Table 4 menunjukkan bahwa halaman yang menggunakan *Query builder* lebih cepat dari pada halaman yang menggunakan *Eloquent*. Pengujian yang terakhir adalah untuk mengetahui halaman mana yang memakan memory lebih banyak. Untuk hasil dari pengujiannya adalah seperti pada Tabel 5

Tabel 5. Hasil Pengujian Memory yang Digunakan Ketika Membuka Halaman

No	*Eloquent*	*Query builder*
1	20 MB	19 MB
2	20 MB	19 MB
3	20 MB	19 MB
4	20 MB	19 MB
5	20 MB	19 MB
6	20 MB	19 MB
7	20 MB	19 MB
8	20 MB	19 MB
9	20 MB	19 MB
10	20 MB	19 MB

Dari data yang ada pada Table 5, dapat disimpulkan bahwa dalam penggunaan *memory*, halaman yang menggunakan *Query builder* lebih hemat 1 MB daripada halaman yang menggunakan *Eloquent*.

Selanjutnya adalah perbandingan *Readability Syntax* dari *Eloquent* dan *Query builder*. Untuk melihat perbandingannya, dapat dilihat perbedaannya dari Gambar 10 dan Gambar 11.

```
102     public function test1(){
103         $karyawans = Karyawan::with('role', 'jabatan')->get();
104         return view('test', compact('karyawans'));
105     }
```

Gambar 10. *Syntax Eloquent*

```
107     public function test2(){
108         $karyawans = DB::table('karyawans')
109         ->join('roles', 'karyawans.role_id', '=', 'roles.id')
110         ->join('jabatans', 'karyawans.jabatan_id', '=', 'jabatans.id')
111         ->select('nama', 'hak_akses', 'nama_jabatan')->get();
112         return view('test2', compact('karyawans'));
113     }
114 }
```

Gambar 11. *Syntax Query builder*

Dapat diamati perbandingan pada kedua *syntax*, bahwa *Eloquent* memiliki penulisan *syntax* yang lebih sederhana, singkat, dan mudah dipahami. Sedangkan pada *Query builder*, penulisan *syntax* terkesan lebih rumit dan sedikit sulit untuk dipahami walaupun output yang ditampilan dari kedua *syntax* tersebut adalah sama. Maka dari itu dalam aspek *Readibility Syntax*, *Eloquent* lebih unggul daripada *Query builder*.
Selanjutnya adalah pembanding terakhir, yaitu perbandingan antara kelebihan dan kekurangan *Eloquent* dan *Query builder*. Dari hasil studi literatur yang penulis lakukan [12] [13], didapatkanlah data kelebihan dan kekurangan masing-masing objek penelitian dan dirangkum pada Table 6 seperti berikut

Tabel 6. Perbandingan Kelebihan dan Kekurangan *Eloquent* dengan *Query builder*

	Eloquent	*Query builder*
Kelebihan	1. Didukung dengan *ORM*, dimana *ORM* memiliki fungsi untuk memetakan data pada sebuah tabel menjadi objek 2. Mendukung perintah CRUD (*create, read, update, delete*) dengan *syntax* kode yang mudah dipahami dan diterapkan 3. Terdapat beberapa *syntax* yang bisa mewakili perintah yang ada pada *Query builder* 4. Menggunakan fungsi *active record* 5. Kode tidak terlalu panjang 6. Bagus untuk projek kecil	1. Menggunakan sedikit memory ketika mengakses halaman 2. Menggunakan sedikit waktu ketika mengakses halaman 3. Cocok digunakan untuk *query* dengan *Big data* 4. Cocok untuk project yang menggunakan *Big data* 5. Cocok digunakan untuk penggunaan fungsi *join* dengan banyak tabel sekaligus

Kekurangan	1. Untuk terhubung dengan tabel di *database*, harus menggunakan perantara Model 2. Bersifat lebih umum 3. Memakan banyak memori 4. Membutuhkan waktu lebih banyak untuk mengeksekusi *query* dan halaman 5. Tidak cocok digunakan untuk *big data* 6. Tidak cocok digunakan jika menggunakan banyak relasi antar tabel	1. Kode *syntax* terkesan rumit dan agak sulit dipahami 2. Beberapa kode terlihat kompleks 3. Tidak cocok untuk proyek-proyek kecil

4. Kesimpulan

Dari hasil dan pembahasan yang telah dilakukan, dapat ditarik kesimpulan bahwa dalam segi performa *Query Builder* lebih unggul daripada *Eloquent*. Dari hasil tes pengujian yang dilakukan, dalam segi kecepatan eksekusi *query* data, didapatkan 9 dari 10 percobaan bahwa waktu eksekusi *query* halaman yang menggunakan *Query Builder* lebih unggul dari pada halaman yang menggunakan *Eloquent*. Dan dari segi kecepatan akses halaman, didapatkan 8 dari 10 percobaan, bahwa waktu kecepatan akses halaman yang menggunakan *Query Builder* lebih cepat dari pada halaman yang menggunakan *Eloqeunt*. Selain itu, dalam aspek penggunaan *memory* ketika mengakses suatu halaman, didapatkan 10 dari 10 percobaan, bahwa halaman yang menggunakan *Query Builder* lebih hemat 1 MB daripada halaman yang menggunakan *Eloquent*. Sedangkan dalam segi kualitas koding, *Eloquent* lebih unggul daripada *Query Builder*. Namun perlu digarisbawahi, bahwa dalam keadaan tertentu, developer *Laravel* harus paham kapan harus menggunakan *Eloquent* dan kapan harus menggunakan *Query Builder*. Sehingga hasil atau produk yang diciptakan memiliki kualitas yang bagus dalam segi *front-end website*, maupun dalam segi *back-end website*. Sehingga dari hasil yang didapatkan, Sistem Manajemen Pekerjaan lebih cocok menggunakan *Query Builder*.

5. Ucapan Terimakasih

Terimakasih disampaikan kepada Bapak Rofiq Azhari S.Kom selaku Kabag Unit SIRS dan Bapak Sutahar, S.T selaku karyawan senior Unit SIRS di Rumah Sakit Muhammadiyah Lamongan yang telah membantu dan membimbing penulis bersama tim dalam perancangan pembuatan Sistem Manajemen Pekerjaan, serta rekan-rekan karyawan Rumah Sakit Muhammadiyah Lamongan.

6. Daftar Pustaka

[1] RSML, "Sejarah Rumah Sakit Muhammadiyah Lamongan," *Sejarah Rumah Sakit Muhammadiyah Lamongan*. https://www.rsmlamongan.com/profil/sejarah-rumah-sakit-muhammadiyah-lamongan (accessed Jan. 10, 2022).
[2] [1] RSML, "Sejarah Rumah Sakit Muhammadiyah Lamongan," Sejarah Rumah Sakit Muhammadiyah Lamongan. https://www.rsmlamongan.com/profil/sejarah-rumah-sakit-muhammadiyah-lamongan (accessed Jan. 10, 2022).
[2] R. Azhari, "Unit SIRS pada RS Muhammadiyah Lamongan," 2021.
[3] Laravel, "Meet Laravel," Installation. https://laravel.com/docs/8.x (accessed Jan. 10, 2022).
[4] Iryana and R. Kawasati, Teknik Pengumpulan Data Metode Kualitatif. Sorong.
[5] M. Nazir, Metode Penelitian. Bogor: Ghalia Indonesia, 2005.
[6] E. Danial and N. Warsiah, Metode Penulisan Karya Ilmiah. Bandung: Laboraturium Pendidikan Kewarganegaraan, 2009.
[7] K. Aikat, "PHP war 1.0: Comparing 5 most popular PHP frameworks, which is the best among all?,"

PHP war 1.0: Comparing 5 most popular PHP frameworks, which is the best among all?, Jul. 12, 2021. https://content.techgig.com/php-war-1-0-comparing-5-most-popular-php-frameworks-which-is-the-best-among-all/articleshow/84344709.cms (accessed Jan. 12, 2021).
[8] Wikipedia, "MySQL," MySQL. 2021. [Online]. Available: https://id.wikipedia.org/wiki/MySQL
[9] The Computer Language Co, "Flowchart," Flowchart. PC Magazine, 2020. Accessed: Jan. 17, 2022. [Online]. Available: https://www.pcmag.com/encyclopedia/term/flowchart#:~:text=A%20graphical%20representation%20of%20the,an%20information%20system%20or%20program.&text=Program%20flowcharts%20show%20the%20sequence,draw%20each%20type%20of%20flowchart.
[10] Y. Waykar and S. S. Mule, "ROLE OF USE CASE DIAGRAM IN S/W DEVELOPMENT," 2015, Accessed: Jan. 17, 2022. [Online]. Available: https://www.researchgate.net/publication/1GV9Jm2u7rmsCe65wKzPTw5jtS38n2tVEGiin_software_development
[11] L. Atencio, "Object-Relational Mapping with Laravel's Eloquent," vol. 15, no. 3, 2016, Accessed: Jan. 17, 2022. [Online]. Available: https://www.phparch.com/wp-content/uploads/2016/03/Object-Relational_Mapping_with_Laravels_Eloquent-phparchitect-March2016.pdf
[12] I. Jound and H. Halimi, "Comparison of performance between Raw SQL and Eloquent ORM in Laravel," 2019.
[13] M. Akash, "Laravel Eloquent Vs DB Query Builder [Performance and other statistics]," 2021.