Analisis Pemanfaatan Laravel Debugbar Dalam Mempermudah Pengerjaan Issue Pada Aplikasi Pusat Pengembangan Sumber Daya Manusia (PPSDM)

Dendy Surya Darmawan
Jurusan Informatika, Fakultas Teknologi Industri
Universitas Islam Indonesia
Yogyakarta, Indonesia
17523078@students.uii.ac.id

Andhik Budi Cahyono
Jurusan Informatika, Fakultas Teknologi Industri
Universitas Islam Indonesia
Yogyakarta, Indonesia
andhikbudi@uii.ac.id

***Abstract*—Dalam pengerjaan *issue* (*bug* dan *error)* pada website aplikasi Pusat Pengembangan Sumber Daya Manusia (PPSDM) bidang Pengadaan Barang dan Jasa (PBJ), *programmer* memerlukan informasi yang mudah dan cepat untuk seperti pencarian nama file *views,*nama *controller,*nama route, dan nama *method controller.* Akan tidak efisien apabila *programmer* melakukan pencarian dengan cara manual. *Packages Laravel Debugbar* merupakan solusi yang digunakan oleh *programmer* proyek PPSDM karena menyediakan ringkasan informasi sebuah halaman mulai dari file *views, route, model, query* dan jenis *collector* lain. *Packages* tersebut menampilkan sebuah panel *overlay* dibagian bawah halaman website. *Packages* tersebut dalam penelitian ini digunakan untuk menangani pengerjaan *issue* proyek PPSDM yaitu mempermudah pencarian nama file *views,* pencarian nama *route* dan *controller,* serta melakukan *print out* sebuah *variable.* Pemanfaatan tersebut kemudian dievaluasi untuk mengetahui apakah *packages* tersebut benar-benar dapat mempermudah pengerjaan *issue* proyek PPSDM. Parameter evaluasi diambil dari subkarakteristik *function suitability* yaitu *function completeness, function correctness, function appropriateness* dan subkarakteristik dari *performance efficiency* yaitu *time behavior* dan *resource utilization* dari model ISO/IEC 25010. Berdasarkan hasil evaluasi yang telah dilakukan, diketahui bahwa *packages Laravel Debugbar* dapat mempermudah dalam pencarian nama file *views,* pencarian nama *route* dan *controller,* serta melakukan *print out* sebuah *variable.*

Keywords—Laravel Debugbar, Analisis, ISO/IEC 25010

I. PENDAHULUAN

A. Latar Belakang

Dalam pengembangan aplikasi berbagai masalah bisa muncul diluar kendali dari tim proyek, pengguna atau klien diharapkan dapat memberikan *feedback* semaksimal mungkin untuk meningkatkan kualitas dari pengembangan aplikasi. Masalah-masalah tersebut biasa disebut dengan istilah *issue.* Adapun jenis *issue* yang dilaporkan dari pengguna atau klien adalah *bug* dan *error* ketika mengakses halaman tertentu.

Pada pengerjaan sebuah *issue* proyek pengembangan sebuah aplikasi, *programmer* tentu harus dapat memahami dengan cepat maksud serta ekspektasi *issue* yang telah dikerjakan, memperkirakan alur pengerjaan serta penulisan kode, mengetahui *boundary* atau batasan kode yang akan digunakan, serta dapat mengimplementasikan kode tersebut dengan kualitas kode yang baik dan mengikuti standar proyek. Kemampuan atau *skills* yang dimiliki *programmer* seiring waktu akan bertambah dengan intensitas pengerjaan berbagai jenis *issue* yang diberikan. Semakin banyak *issue* yang dikerjakan maka tentu semakin tinggi tantangan yang akan datang serta menimbulkan persepsi dari *programmer* bahwa ada kemungkinan kegagalan dalam mengerjakan *issue* tersebut. Beberapa penelitian mengungkapkan bahwa tantangan dalam mengerjakan sesuatu hal secara positif dapat meningkatkan motivasi dalam diri. Dalam penelitian yang dilakukan oleh Karina Wilkie tahun 2016, menemukan bahwa siswa yang menerima tantangan lebih dalam *issue* matematika maka secara positif akan memberikan dampak dua sisi yaitu (kesenangan, keterikatan, ketertarikan) dan manfaat pembelajaran [1]. Selain itu dampak lain yang ditemukan adalah secara efektif meningkatkan motivasi siswa, performa pembelajaran, dan rasa kepuasan dalam pembelajaran [2]. Dengan kata lain *programmer* memang akan menghadapi tantangan yang sulit, namun dampak yang didapatkan selama mengerjakan sebuah *issue* hingga menyelesaikannya tentunya akan bermanfaat untuk peningkatan kualitas dan kemampuan dalam bekerja.

Meningkatkan kemampuan dengan menambah intensitas dalam mengerjakan *issue* memang menjadi kewajiban bagi *programmer*, karena dari penugasan tersebut *programmer* akan mendapatkan lebih banyak pengalaman serta menyerap lebih banyak ilmu. Namun kemampuan yang dimiliki oleh setiap *programmer* masih belum cukup. *Programmer* perlu menggunakan *tools* yang relevan dan tepat agar pekerjaan membantu dalam bekerja. Ada berbagai macam jenis tools yang membantu dalam pengembangan sebuah perangkat lunak atau aplikasi dengan fungsi-fungsi tertentu seperti *tools communication team, database environment, code editor, task collaboration tools, version control systems* dan masih banyak *tools* lain. Salah satu *tools* yang digunakan dalam proyek yang diangkat pada makalah ini yaitu aplikasi PPSDM (Pusat Pengembangan Sumber Daya manusia) bidang PBJ (Pengadaan Barang/Jasa) adalah *packages Laravel Debugbar.*

Laravel Debugbar merupakan sebuah *packages Laravel* yang membantu *programmer* dengan mengumpulkan data-data menggunakan kelas PHP yaitu *DataCollector*[3] yang kemudian menjadi sebuah informasi yang relevan, valid, akurat dan lengkap yang dapat digunakan dalam proses *debugging* atau menulis kode. Cara kerja hingga keputusan yang dibuat oleh *programmer* salah satunya berpengaruh karena informasi yang baik dan berkualitas[4]. Beberapa pemanfaatan *packages* ini dalam proses pengerjaan *issue* pada proyek PPSDM yaitu memberikan kemudahan dalam pencarian nama file *views, route* dan *controller* yang digunakan serta juga dapat melakukan *print out* sebuah *variable* sebagai pengganti *function* umum di Laravel yaitu *dump and die* atau *dd(). Packages* ini berbentuk sebuah panel dibagian bawah layar aplikasi, dengan masing-masing *tab* sesuai dengan *collectors* yang telah disediakan *packages* ini

seperti *views, messages, exceptions, query, route, models, cache* dan *collector* lainnya.

Pada makalah ini akan dibahas tentang analisis pemanfaatan *packages Laravel Debugbar* dalam pengerjaan *issue* proyek PPSDM yang diukur dengan menggunakan parameter pada dimensi *product quality* model ISO/IEC 25010. Parameter yang digunakan ada 5 yang dibagi menjadi 3 subkarakteristik dari *function suitability* yaitu *functional completeness, functional correctness* dan *functional appropriateness.* Sedangkan 2 subkarakteristik selanjutnya dari *performance efficiency* yaitu *time behavior* dan *resource utilization* [5]. Model ISO/IEC 25010 merupakan versi lanjutan dari ISO/IEC 9126 yang telah direvisi secara teknis menjadi standar internasional terbaru dalam mengevaluasi kualitas perangkat lunak.

B. *Rumusan Masalah*

Berikut merupakan rumusan masalah dari makalah tentang analisis pemanfaatan *Laravel Debugbar* dalam mempermudah pengerjaan *issue* sebagai berikut.

1) Bagimana menganalisis manfaat *Laravel Debugbar* untuk mempermudah pengerjaan *issue* pada proyek PPSDM dengan menggunakan parameter evaluasi *function suitability* dan *performance efficiency* dari model ISO/IEC 25010?

C. *Tujuan Penulisan*

Berikut ini tujuan yang ingin dicapai dari penulisan makalah sebagai berikut.

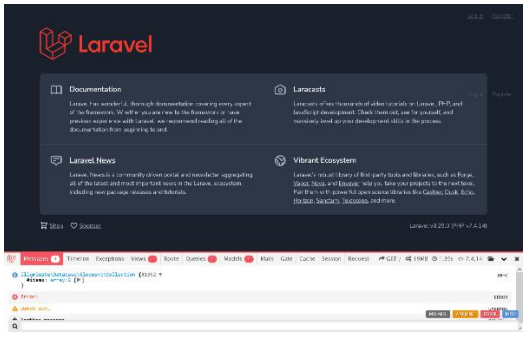
1) Untuk mengetahui hasil analisis manfaat *Laravel Debugbar* dalam mempermudah pengerjaan *issue* dalam proyek PPSDM dengan menggunakan parameter *evaluasi function suitability* dan *performance efficiency* dari model ISO/IEC 25010.

II. KAJIAN PUSTAKA

A. *Landasan Teori*

1) Laravel Debugbar

Laravel Debugbar merupakan sebuah packages dari PHP Debugbar yang terintegrasi dengan framework Laravel[6]. Packages ini dibuat oleh seorang Web Developer sekaligus Co-Founder dan Lead Developer FruitCake asal Belanda yakni Barry van de Heuvel. *Packages Laravel Debugbar* menggunakan salah satu kelas PHP yaitu DataCollectors, kelas ini berfungsi untuk mengumpulkan data-data menjadi sebuah informasi baru. Packages ini akan menampilkan informasi yang sudah dikumpulkan pada panel *overlay* dibagian bawah *website* dengan masing-masing *tab* dengan jenis *collectors* tertentu seperti pada Gambar 1.



Gambar 1. Tampak tampilan Packages Laravel Debugbar

Adapun *default collector* dari *packages ini* sebagai berikut.

- PhpInfoCollector, menampilkan versi PHP yang sedang digunakan aplikasi.
- MessagesCollector, menampilkan pesan dengan *method* seperti *error, warning,* info. Collector ini dapat pula melakukan kostumisasi pesan. Selain menampilkan pesan fungsi lain *collector* ini adalah sebagai tempat *output* print sebuah *variable* menggantikan *function* umum Laravel yaitu *dump and die* atau *dd().*
- TimeDataCollector, menampilkan waktu yang dibutuhkan untuk *load* semua data di halaman (*booting* dan *application.*
- *MemoryCollector,* menampilkan penggunaan *memory* yang dihabiskan untuk menjalankan setiap halaman.
- *ExceptionsCollector,* menampilkan error *exception* (perubahan alur program dari kondisi normal ke kondisi tertentu). Ketika aplikasi sedang berjalan, error ini tidak akan menyebabkan *crash* atau aplikasi berhenti melainkan aplikasi tersebut akan berjalan tidak sesuai dengan ekspektasi, *exception* biasanya terjadi ketika adanya transaksi data ke *database*.

Packages Laravel Debugbar juga memiliki *collector* lain seperti views, messages, exceptions, query, route, models, cache dan masih banyak collector lainnya.

2) Pemanfaatan Packages Laravel Debugbar Dalam Pengerjaan Issue Proyek PPSDM

Issue yang dikerjakan oleh *programmer* merupakan masalah yang ditemukan oleh klien atau pengguna pada saat menjalankan aplikasi setelah rilis versi terbaru aplikasi. Adapun *issue* yang dikerjakan oleh peneliti yaitu pada saat bergabung dengan proyek PPSDM sebagai berikut.

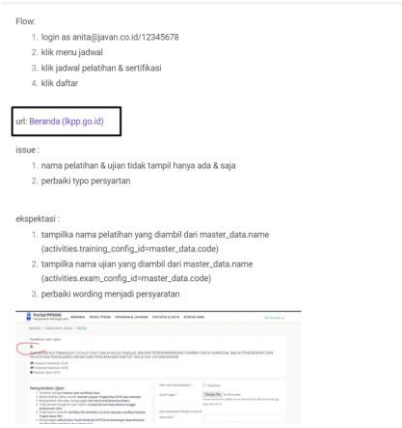
- *Bug,* sebuah kesalahan dimana perangkat lunak tidak menjalankan seperti yang seharusnya dilakukan begitupun sebaliknya. Contoh seperti ikon pencil yang digunakan untuk menu *edit* tidak muncul.
- *Error,* sebuah kesalahan pada aplikasi yang menyebabkan aplikasi tidak dapat menjalankan fungsinya. Contoh ketika mengakses halaman tertentu muncul error misal "*class not found*".

Adapun prosedur atau tahapan untuk menyelesaikan sebuah *issue* proyek PPSDM sebagai berikut.

1. Memahami ekspektasi *issue* yang dikerjakan
2. Menulis kode baru atau *refactoring* kode lama
3. Melakukan *commit, push kode* dan *merge request*
4. Melakukan pengecekan *gitpipeline* setelah *merge request*
5. Menunggu hasil *review* kode *merge request* oleh *senior programmer*
6. Mengubah *label* tugas *menjadi ready to test* dan menunggu hasil pengujian oleh tester
7. Tugas selesai.

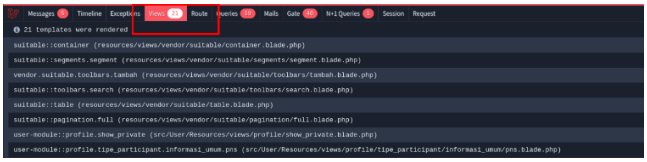
Pemanfaatan *packages Laravel Debugbar* terjadi pada tahapan pertama dan kedua prosedur penyelesaian *issue* diatas. Berikut ini pemanfaatan *packages Laravel Debugbar* pada tahapan tersebut.

- Pencarian nama file *views*, *route* dan *controller*. Tim PPSDM menggunakan *ActiveCollab* sebagai *tools* dalam mengatur manajemen proyek salah satunya adalah pembagian *issue*. Pada setiap *issue* yang diberikan kepada *programmer* terdapat lampiran berupa *url* sebagai penanda bahwa halaman tersebut terdapat sebuah *issue* ataupun halaman yang akan ditambah fitur baru. Pada Gambar 2 merupakan contoh lampiran *url* pada *issue* yang diberikan.



Gambar 2. Contoh issue ActiveCollab proyek PPSDM

Setelah mengakses *url* tersebut, *programmer* dapat memulai mengerjakan *issue* yang diberikan, langkah pertama yang harus dilakukan adalah mengetahui nama file *views* yang digunakan pada halaman tersebut. Dengan menggunakan *packages* Laravel Debugbar informasi tersebut dapat dengan mudah didapatkan yaitu pada *tab views*, *tab* tersebut akan menampilkan informasi semua nama-nama file *views* serta jumlah yang di-*load* pada halaman yang sedang diakses seperti pada Gambar 3. Sedangkan informasi mengenai nama *route* dan *controller* yang digunakan ada pada *tab Route.* Catatan untuk penggunaan *packages* ini tidak selalu digunakan untuk mengakses halaman yang terdapat *issue*, dapat digunakan untuk mendapatkan informasi halaman lain.



Gambar 3. Contoh tab views package Laravel Debugbar

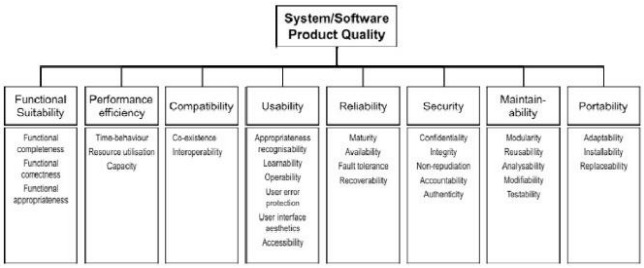
- Melakukan print out sebuah variable dalam pengecekan output baris kode. Dalam menulis sebuah baris kode perlu dipastikan bahwa kode menghasilkan *output* yang benar sesuai ekspektasi yang diharapkan. Salah satu cara pengecekan baris kode menghasilkan *output* yang benar adalah dengan melakukan *print* out sebuah *variable* dari baris kode yang ditulis tersebut secara langsung. *Framework Laravel* menyediakan *function dump and die* atau 'dd()' yang berfungsi untuk menghasilkan *output* dari *variable* namun memberhentikan eksekusi kode selanjutnya. Pada *packages Laravel Debugbar* juga dapat melakukan *print variable* tanpa memberhentikan eksekusi kode yang berjalan yaitu dengan memasukkan *variable* tersebut ke dalam *façade class* dari *Laravel Debugbar* atau menggunakan metode *non-static* dengan perintah pada baris ke-4 dan baris ke-9 seperti Gambar 4.

```
1   $user = auth()->user();
2
3   //façade Laravel Debugbar/static method
4   Debugbar::nama_properti_atau_method('nama_variabel')
5   //contoh
6   Debugbar::info($user);
7
8   //non-static method
9   app('debugbar')->nama_properti_atau_method('nama_variabel');
10  //contoh
11  app('debugbar')->info($user);
12
```

Gambar 4. Notasi print out sebuah variable

3) Model ISO/IEC 25010

ISO/IEC merupakan standar dunia internasional dalam mengevaluasi kualitas sebuah perangkat lunak yang dikembangkan oleh *International Organization for Standarization and International Electronical Commission* disingkat dengan ISO/IEC. Adapun versi yang digunakan pada makalah ini adalah versi ISO/IEC 25010 edisi pertama tahun 2011, pengembangan dari versi ISO/IEC 9126 tahun 2001 yang telah direvisi secara teknis, menjadi standar internasional terbaru dan relevan untuk menguji softaware dan sistem komputer yang dikembangkan. ISO/IEC 25010 memiliki 8 karakteristik yang menjadi tolak ukur dalam mengevaluasi kualitas perangkat lunak berdasarkan dimensi *product quality* yaitu *functional suitability*, *performance efficiency*, *compatibility*, *usability*, *reliability*, *security*, *maintainability*, dan portability seperti pada Gambar 5[5].



Gambar 5. Model product quality pada ISO/IEC 25010

4) Analisis Pemanfaatan Packages Laravel Debugbar dengan ISO/IEC 25010

Menurut Peter Salim dan Yenni Salim (2002), analisis merupakan proses pemecahan masalah yang diawali dengan membuat sebuah hipotesis kemudian melakukan pengamatan dan percobaan untuk membuktikan kebenaran dari hipotesis tersebut[7]. Adapun langkah yang digunakan untuk membuktikan hipotesis adalah dengan melakukan evaluasi pemanfaatan *packages Laravel Debugbar* pada saat pengerjaan *issue* proyek PPSDM dengan parameter pada Model ISO/IEC 25010.Parameter yang digunakan sebagai pada makalah ini yaitu 5 parameter, yaitu 3 subkarakteristik dari karakteristik *function suitability* ada *functional completeness*, *functional correctness*, *functional appropriateness.* Sedangkan karakteristik *performance efficiency* terdapat 2 subkarakteristik yaitu *time behavior*, dan *resource utilization* (**Error! Reference source not found.**). Terkait penjelasan lengkap dari 5 subkarakteristik dan 2 karakteristik tersebut sebagai berikut.

Function suitability adalah kemampuan sebuah sistem dalam menyediakan fungsi yang dapat memenuhi kebutuhan tertentu untuk digunakan di kondisi tertentu. Subkarakteristik yang digunakan pada makalah sebagai evaluasi *packages Laravel Debugar* ada 3 sebagai berikut.

- **Functional Completeness**, kondisi dimana suatu sistem dapat mencakup semua *issue* yang ditentukan dan memenuhi tujuan dari pengguna.
- **Functional Correctness**, kondisi dimana suatu sistem menyediakan hasil yang benar dengan tingkat presisi yang diperlukan.
- **Functional Appropriateness**, kondisi dimana suatu sistem dapat memfasilitasi pencapaian *issue* dan tujuan tertentu.

Performance efficiency adalah kemampuan sebuah sistem dengan kinerja relatif terhadap jumlah sumber daya yang digunakan dalam kondisi yang ditentukan. Subkarakteristik yang digunakan pada makalah sebagai evaluasi *packages Laravel Debugar* ada 2 sebagai berikut..

- **Time Behavior**, tingkat dimana respon dan waktu proses suatu produk atau sistem dalam memenuhi kebutuhan tertentu.
- **Resource Utilization**, tingkat dimana jumlah dan jenis sumber daya yang digunakan oleh suatu produk atau sistem ketika menjalankan fungsinya memenuhi persyaratan.

Adapun pemilihan 2 dari 8 karakteristik dari dimensi *product quality* pada ISO/IEC 25010 karena paling cocok untuk dapat mengevaluasi pemanfaatan *packages Laravel* yaitu segi fungsionalitas dan efisiensi performa. Selain itu evaluasi ini hanya untuk menilai kualitas *packages Larvel* bukan sebuah sistem perangkat lunak.

B. *Hasil Penelitian Yang Relevan*

Penelitian pertama menjelaskan penggunaan model ISO/IEC 25010 sebagai model untuk mengevaluasi kualitas sistem *e-government* setelah pengimplementasian *framework Laravel.* Hasil penelitian menunjukkan bahwa pengujian dengan *black box testing* model tersebut memiliki nilai *function suitability, reliability, usability, performance efficiency, maintainability, dan portability* yang cukup baik dan sesuai harapan[8]. Penelitian kedua menjelaskan model ISO/IEC 25010 dapat digunakan untuk mengevaluasi kualitas aplikasi android First Kid. Penelitian tersebut menghasilkan nilai rata-rata dari masing-masing karakteristik yang digunakan sebagai acuan dalam memetakan karakteristik yang perlu dilakukan peningkatan kualitas[9].

Dari dua Penelitian tersebut terdapat perbedaan yaitu subkarteristik model yang digunakan dan teknik pengujian yang dilakukan. Hal ini mendasari peneliti bahwa model ISO/IEC 25010 dapat menggunakan beberapa subkarateristik tertentu dan teknik pengujian dapat dilakukan sesuai keperluan tidak harus menggunakan kuesioner atau *black box testing.*

III. METODOLOGI

Penyusunan makalah ini dibuat dengan metodologi yaitu, 1) evaluasi pemanfaatan *packages Laravel Debugbar* dalam pencarian nama file *views*, pencarian nama *route* dan *controller*, serta penggunaan *packages* ini dalam melakukan *print out* sebuah *variable.* Parameter yang digunakan berdasarkan masing-masing subkarakteristik *function suitability* dan *performance efficiency* ISO/IEC 25010. 2) penarikan kesimpulan setelah evaluasi.

Adapun penilaian dan bobot skor untuk parameter evaluasi subkarateristik model ISO/IEC 25010 berdasarkan asumsi dari peneliti selama menggunakan *packages Laravel Debugbar* dalam pengerjaan *issue* proyek PPSDM. Pada Table III.1 merupakan penilaian dan bobot skor evaluasi.

Table III.1 Penilaian dan bobot skor evaluasi parameter subkarateristik ISO/IEC 25010

Pernyataan	***Bobot Skor***
Sangat Baik (SB)	3
Baik (B)	2
Kurang Baik (KB)	1

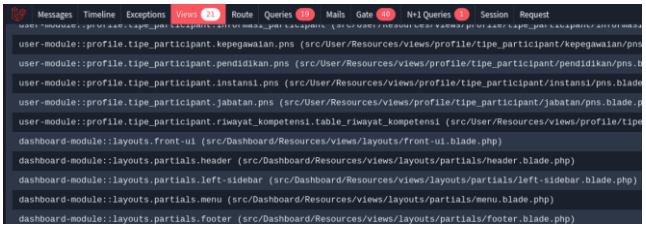
IV. PEMBAHASAN

A. Evaluasi *packages Laravel Debugbar*

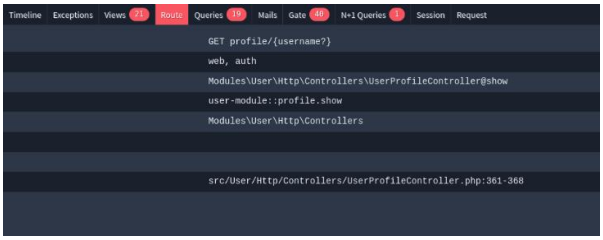
1) Functional Completeness, kemampuan *packages* untuk menyediakan kebutuhan tertentu dan memenuhi tujuan yang diinginkan dari *programmer.*

a) Pencarian nama file *views*, *route* dan *controller*

Evaluasi berdasarkan *functional completeness, packages Laravel Debugbar* dapat menyediakan informasi yang dibutuhkan *programmer* terkait nama file *views*, *route* dan *controller* yang digunakan (*Gambar 6* dan *Gambar 7*). Hal ini mempermudah *programmer* karena tidak perlu mencari secara manual informasi tersebut, belum lagi ada halaman yang menggunakan *blade extend* (penggunaan kode program lain pada kode program lainnya) pada *Gambar 6*.



Gambar 6 Tampilan nama-nama file *views* yang mengimpelentasikan *blade extend* pada halamannya

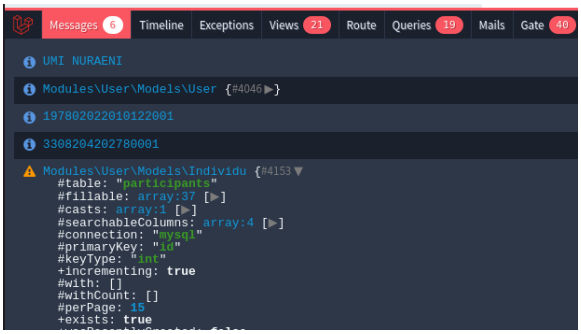


Gambar 7 Tampilan *tab Route*

b) Melakukan *print out* sebuah *variable*

Evaluasi berdasarkan *functional completeness, packages Laravel Debugbar* dapat melakukan *print* out

sebuah *variable* dengan *output* pada *tab messages*, yang digunakan untuk melakukan pengecekan *output* sebuah *variable* dalam baris kode terdahulu atau menguji *variable* yang ingin diimpelementasikan (*Gambar 8*).



Gambar 8. Tampilan print out variable

2) Functional Correctness, kemampuan *packages* untuk menyediakan hasil yang benar sesuai dengan tingkat presisi yang diperlukan.

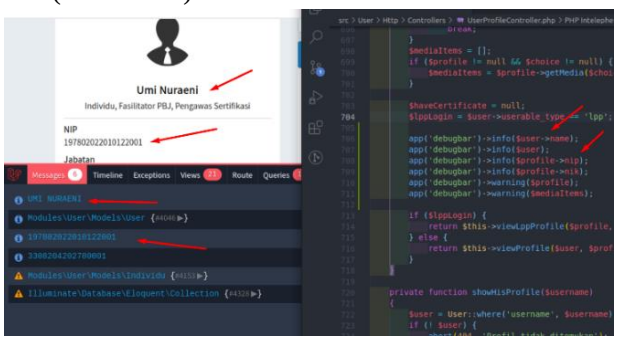
a) Pencarian nama file *views*, *route* dan *controller*

Evaluasi berdasarkan *functional correctness*, *packages Laravel Debugbar* menyediakan hasil data yang benar dan valid mengenai letak baris kode dari *method controller* yang digunakan pada halaman tersebut (*Gambar 9*).

Gambar 9. Letak baris *method controller* sesuai berdasarkan informasi dari *tab Route*

b) Melakukan *print out* sebuah *variable*

Evaluasi berdasarkan *functional correctness*, *packages Laravel Debugbar* menampilkan hasil data yang benar serta valid terkait *print out variable* pada *tab messages* (*Gambar 10*).

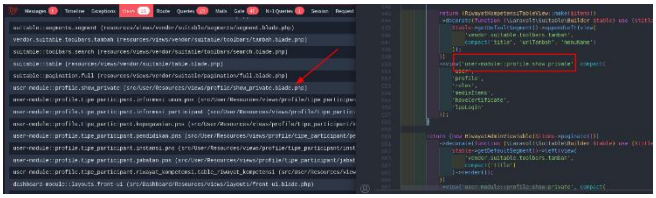


Gambar 10. Data valid pada saat *print out variable*

3) Functional Appropriateness, kemampuan *packages* untuk memfasilitasi pencapaian *issue* dan tujuan tertentu.

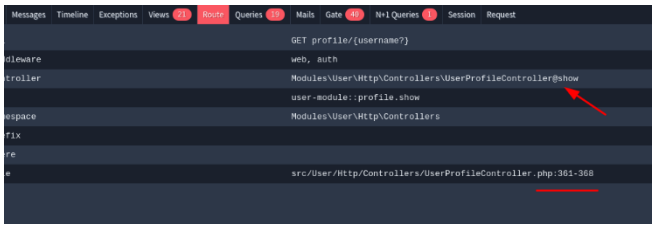
a) Pencarian nama file *views*, *route* dan *controller*

Evaluasi berdasarkan *functional appropriateness*, pencarian nama file *views* menggunakan pada *tab views* hanya menampilkan informasi nama file *views* yang di-*load*, tetapi tidak dapat memfasilitasi satu nama file *views* yang menjadi *base* html yang digunakan pada setiap halaman yang diakses (*Gambar 11*).



Gambar 11. Tidak memfasilitasi *base html* yang digunakan di halaman tersebut

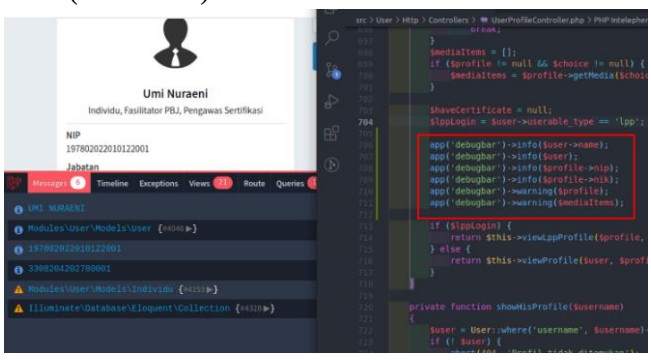
Sedangkan pencarian nama *route* dan *controller* ini juga memfasilitasi *programmer* berupa informasi berupa letak baris dari *method controller* yang digunakan pada halaman tersebut (*Gambar 12*). Selain itu juga *packages* ini menampilkan informasi spesifik seperti nama *uri*, jenis *middleware*, nama *prefix*, dan nama file.



Gambar 12. Informasi tambahan pada *tab Route* berupa letak baris kode *method controller*

b) Melakukan *print out* sebuah *variable*

Evaluasi berdasarkan *functional appropriateness*, *packages Laravel Debugbar* dapat melakukan *multiple print out* beberapa *variable*. Hal ini tentu bermanfaat agar *programmer* dapat mengujikan beberapa *variable* sekaligus yang telah dibuat maupun melakukan pengecekan *output variable* baris kode terdahulu (*Gambar 13*).



Gambar 13. Multiple print out beberapa *variables*

4) Time Behavior, kemampuan *packages* dalam menampilkan respon dan waktu proses suatu produk atau sistem dalam memenuhi kebutuhan tertentu.

a) Pencarian nama file *views*, *route* dan *controller*

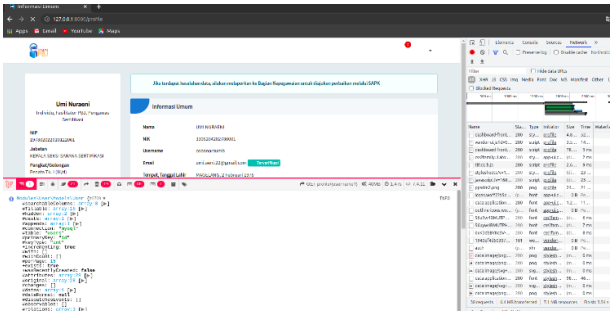
Evaluasi berdasarkan *time behavior*, *packages Laravel Debugbar* memang akan ada tambahan waktu respon

dan waktu proses karena melakukan *request* data ke semua *collector*, namun hal ini lebih baik dari pada melakukan pencarian secara manual. Berikut ini perbandingan langkah pencarian nama file *views*, *route* dan *controller* menggunakan *packages Laravel Debugbar* dan secara manual.

- *Packages Laravel Debugbar*: 1) buka *tab views* dan *tab route*. 2) akses halaman berdasarkan informasi dari 2 *tab* tersebut.
- Tanpa *packages Laravel Debugbar*: 1) buka file '*web.php*'. 2) cari *route* yang menggunakan *uri* halaman yang sedang diakses. 3) buka *method controller* dari *route* yang sudah ditemukan. 4) buka file *views* dengan nama sesuai dengan dalam perintah *return view('nama_file')* pada *method controller*.

b) Melakukan *print out* sebuah *variable*

Evaluasi berdasarkan *time behavior, packages Laravel Debugbar* dalam melakukan *print out* sebuah *variable* waktu respon yang didapatkan yaitu 3,54s (*Gambar 14*), tidak berbeda jauh jika dibandingkan *print out* menggunakan *function dump and die* atau *dd()* yaitu 1,71s (*Gambar 15*). Perbedaan ini terjadi karena *packages Laravel Debugbar* melakukan *request* data ke semua *collector* hingga menampikan panel, sedangkan *print out* menggunakan *dd()* akan memberhentikan semua proses eksekusi setelah baris perintah *print out* dijalankan.



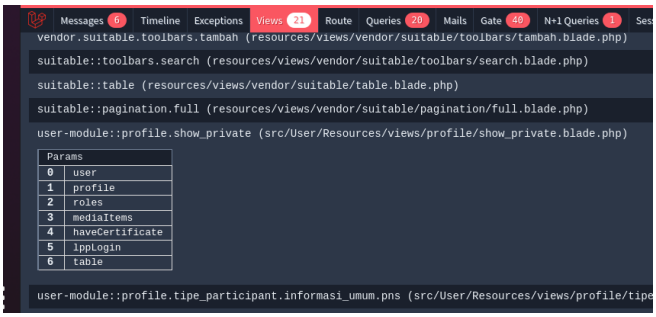
Gambar 14. Tampilan waktu respon *print out* sebuah *variable* menggunakan *packages Laravel Debugbar*

Gambar 15. Tampilan waktu respon print out sebuah variable menggunakan function dd()

5) Resource Utilization, kemampuan *packages* Laravel Debugbar dalam memanfaatkan jumlah dan jenis sumber daya tertentu ketika sedang menjalankan fungsinya memenuhi persyaratan.

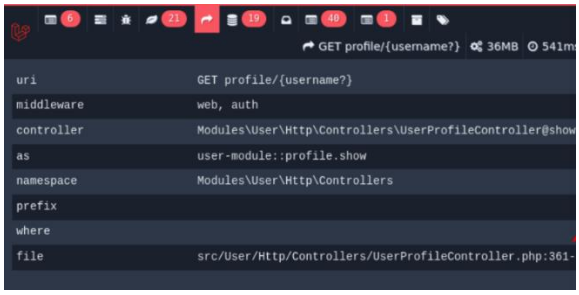
a) Pencarian nama file *views*, *route* dan *controller*

Evaluasi berdasarkan *resource utilization*, pencarian nama file *views* pada *tab views* memberikan informasi tambahan berupa parameter yang dikirimkan pada masing-masing file *views* (*Gambar 16*).



Gambar 16. Parameter yang dikirimkan ke *views*

Sedangkan pencarian nama *route* dan *controller* pada *tab route*, terdapat pembagian secara spesifik informasi seperti *uri*, jenis *middleware*, nama *prefix*, nama file, nama *method controller* serta letak barisnya (*Gambar 17*).



Gambar 17. Informasi spesifik pada *tab Route*

b) Melakukan *print out* sebuah *variable*

Evaluasi berdasarkan *resource utilization, packages Laravel Debugbar* tidak ada pemanfaatan informasi tertentu pada *tab messages* jika melakukan *print out* sebuah *variable*.

B. Hasil evaluasi *packages Laravel Debugbar*

Berikut ini tabel dan penilaian yang sudah dirangkum dari evaluasi parameter subkarateristik *function suitability* dan *performance efficiency* ISO/IEC 25010.

Table IV.1Hasil evaluasi pemanfaatan packages Laravel Debugbar dalam pengerjaan issue proyek PPSDM

Parameter Evaluasi	**Pemanfaatan Packages Laravel Debugbar dalam proyek PPSDM**		
	Pencarian nama file views	*Pencarian route & controller*	*Melakukan print out sebuah variable*
Functional Completness	Sangat Baik	Sangat Baik	Baik
Functional Correctness	Sangat Baik	Sangat Baik	Baik
Functional Appropriateness	Kurang baik	Sangat Baik	Sangat Baik
Time Behavior	Sangat Baik	Sangat Baik	Baik
Resource Utilization	Baik	Sangat Baik	Kurang baik

Parameter Evaluasi	Pemanfaatan Packages Laravel Debugbar dalam proyek PPSDM		
	Pencarian nama file views	*Pencarian route & controller*	*Melakukan print out* sebuah *variable*
Total Skor	12	15	10
Rata-rata	2,4 (Baik)	3 (Sangat Baik)	2 (Baik)

1) Hasil evaluasi berdasarkan functional completeness

a) Pencarian nama file *views* menggunakan *packages Laravel Debugbar* dianggap **sangat baik** dengan asumsi mampu memenuhi kebutuhan *programmer* untuk mencari informasi terkait nama file *views* yang digunakan pada halaman.

b) Pencarian nama *route* dan *controller* menggunakan *packages Laravel Debugbar* dianggap **sangat baik** dengan asumsi mampu memenuhi kebutuhan *programmer* untuk informasi nama *route* dan *controller* yang digunakan pada sebuah halaman.

c) Melakukan *print out* sebuah variable dianggap **baik** dengan asumsi dapat memenuhi kebuthan *programmer* untuk melakukan *print out* sebuah *variable* dengan *output* pada *tab messages.*

2) Hasil evaluasi berdasarkan *functional correctness*

a) Pencarian nama file *views* dianggap **sangat baik** dengan asumsi memberikan hasil data yang valid semua nama file *views* yang ada pada *tab views*.

b) Pencarian nama *route* dan *controller* dianggap **sangat baik** dengan asumsi memberikan data yang valid terkait nama *route* dan *controller* yang digunakan pada halaman yang diakses.

c) Melakukan *print out* sebuah *variable* dianggap **baik** dengan asumsi memberikan hasil data yang valid *output print out* pada *tab messages.*

3) Hasil evaluasi berdasarkan *functional appropriateness*

a) Pencarian nama file *views* dianggap **kurang baik** dengan asumsi tidak secara spesifik memberikan informasi terkait nama file *view* yang menjadi *base html*, sehingga perlu mencari nama file secara manual pada halaman yang diakses.

b) Pencarian nama *route* dan *controller* dianggap **sangat baik** dengan asumsi memfasilitasi informasi tambahan dengan memberikan spesifik letak baris kode dari *method controller*.

c) Melakukan *print out* sebuah *variable* dianggap **sangat baik** dengan asumsi menyediakan *multiple print* beberapa *variable* dengan *output* tersusun sistematis pada *tab messages* dan tidak menghentikan program berjalan.

4) Hasil evaluasi berdasarkan *time behavior*

a) Pencarian nama file *views* dianggap sangat **sangat baik** dengan asumsi lebih mudah dan cepat menemukan semua nama file *views* pada *tab views* dibandingkan dengan cara manual.

b) Pencarian nama *route* dan *controller* dianggap **sangat baik** dengan asumsi lebih mudah dan cepat menemukan nama *route* dan *controller* pada *tab route* dibandingkan dengan cara manual.

c) Melakukan *print out* sebuah *variable* dianggap **baik** dengan asumsi waktu respon dan waktu proses tidak terlalu jauh signifikan perbedaannya dibandingkan melakukan *print out* dengan *function* umum *dump and die* atau dd().

5) Hasil evaluasi berdasarkan *resource utilization*

a) Pencarian nama file *views* dianggap **baik** dengan asumsi menyediakan informasi tambahan berupa parameter yang dikirimkan setiap file *views* pada *tab messages.*

b) Pencarian nama *route* dan *controller* dianggap **sangat baik** karena menyediakan informasi spesifik seperti nama *uri*, jenis *middleware*, nama *prefix*, dan letak baris kode dari *method controller* pada *tab route.*

c) Melakukan *print out* sebuah *vairable* dianggap **kurang baik** karena tidak ada pengolahan sumber daya dari untuk menyediakan informasi baru pada *tab messages.*

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Adapun hasil kesimpulan yang diperoleh serta korelasinya dengan hipotesis adalah sebagai berikut.

1) Dari tabel hasil evaluasi menunjukkan rata-rata skor dari tiga pemanfaatan *packages Laravel Debugbar*, dua pemanfaatan memiliki nilai baik yaitu pencarian nama file *views* dan melakukan *print out* sebuah *variable*. Sementara satu pemanfaatan yaitu pencarian nama *route* dan *controller* memiliki nilai sangat baik. Dapat disimpulkan bahwa *packages Laravel Debugbar* dapat membantu mempermudah pengerjaan *issue* proyek PPSDM.

B. Saran

Untuk mendapatkan hasil penelitian yang lebih baik, berikut ini saran yang harus dilakukan untuk penelitian selanjutnya.

1) Memiliki skala penilaian yang jelas dan dapat diukur bukan hanya sekadar asumsi sehingga menghasilkan kesimpulan benar terbukti valid tidak menimbulkan ambiguitas.

2) Melanjutkan penelitian dengan lebih terkonsep dan terstruktur tepatnya pada instrumen penelitian, perhitungan data, pengujian data serta penarikan kesimpulan agar hipotesis kedua terbukti secara matematis.

REFERENSI

[1] K. Wilkie, "Rise or Resist: Exploring Senior Secondary Students' Reactions to Challenging Mathematics Tasks Incorporating Multiple Strategies," *Eurasia J. Math. Sci. Technol. Educ.*, vol. 12, pp. 2061–2083, 2016.

[2] C.-Y. Hung, J. C.-Y. Sun, and P.-T. Yu, "The benefits of a challenge: student motivation and flow experience in tablet-PC-game-based learning," *Interact. Learn. Environ.*, vol. 23, pp. 172–190, 2015.

[3] Symfony, "How to Create a custom Data Collector (Symfony Docs)," *2020*. https://symfony.com/doc/current/profiler/data_collector.html (accessed Jun. 01, 2021).

[4] V. Rivai, *Manajemen Sumber Daya Manusia untuk Perusahaan: Dari Teori Ke Praktek.* Jakarta: PT Rajagrafindo Persada, 2009.

[5] ISO/IEC, “ISO / IEC 25010 : 2011 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models,” 2011.

[6] B. vd. Heuvel, “barryvdh/laravel-debugbar:Laravel Debugbar - Github,” 2021. https://github.com/barryvdh/laravel-debugbar (accessed May 08, 2021).

[7] P. Salim and Y. Salim, *Kamus bahasa Indonesia kontemporer*. Jakarta: Modern English Press, 2002.

[8] S. Rahayuda, “Evaluasi Penggunaan Framework Laravel pada E-government Menggunakan ISO/IEC 25010:2011,” *J. Ilmu Pengetah. dan Teknol. Komun.*, vol. 19, no. 1, pp. 81–94, 2017.

[9] M. Harun, “EVALUASI KUALITAS PERANGKAT LUNAK DENGAN ISO/IEC 25010: 2011 (STUDY KASUS: APLIKASI FIRST AID PADA PLATFORM ANDROID),” *J. Akrab Juara*, vol. 3, pp. 53–61, 2018.