Evaluasi Penggunaan *Framework Laravel* Pada *E-government* Menggunakan ISO/IEC 25010:2011

Evaluation of Laravel Framework on E-government Using ISO/IEC 25010:2011

I Gede Surya Rahayuda
STMIK STIKOM Bali, Jl. Raya Puputan Renon No. 86, Denpasar

surya.rahayuda@gmail.com

Naskah diterima: 15 Desember 2016, direvisi: 6 Juli 2017 disetujui: 17 Juli 2017

Abstrak

Evaluasi penggunan teknologi informasi diperlukan untuk mengidentifikasi atau memeriksa seberapa berhasil sistem yang telah dibuat atau diimplementasikan. Penelitian ini merupakan evaluasi dari simulasi implementasi yang telah dibangun sebelumnya. Simulasi teknologi informasi dilakukan pada salah satu sistem e-government pada bidang kesehatan, yaitu EJKBM. Sistem dibangun menggunakan metode SDLC, alur program dan basis data yang dibuat menggunakan ERD, DFD dan MVC. Pengujian sistem dilakukan menggunakan white box dan black box testing dan evaluasi dilakukan dengan menggunakan ISO/IEC 25010:2011. Berdasarkan nilai cyclomatyc complexity V(G) pada pengujian white box sistem dinyatakan relevan, pada pengujian black box didapatkan hasil bahwa sistem sesuai dengan hasil yang diharapkan dan berdasarkan pengujian menggunakan ISO/IEC 25010:2011 sistem dinyatakan memiliki nilai fungsionalitas, kehandalan, penggunaan, efisiensi, pemeliharaan dan portabilitas yang cukup baik.

Kata Kunci: evaluasi, *laravel*, *white box test, black box test, ISO/IEC-25010:2011*

Abstract

Evaluation of the use of information technology is needed to identify or check how successful the system has been created or implemented. This research is an evaluation of the simulation implementation that has been built previously. Simulation of information technology is done on one of the e-government system in health field that is EJKBM. The system was built using the SDLC method, program flow and database was created using ERD, DFD and MVC. System testing is done using white box and black box testing. The evaluation is done by using ISO/IEC 25010:2011. Based on the value of cyclomatyc complexity V (G) on test white box system stated relevant, on testing black box obtained results that the system in accordance with the expected results and based on testing using ISO / IEC 25010: 2011 system stated has a good functionality, reliability, usability, efficiency, maintainability and portability value.

***Keyword**: evaluation, laravel, white box test, black box test, ISO/IEC-25010:2011*

PENDAHULUAN

Penelitian ini merupakan lanjutan dari penelitian sebelumnya. Oleh karena itu, pada pembahasan ini tidak terlalu banyak memberikan penjelasan mengenai pengertian dari *electronic government,* elektronik jaminan kesehatan bali mandara dan juga mengenai *framework laravel* (Rahayuda 2017) (Susanto 2016). Pada pembahasan kali ini akan lebih banyak menjelaskan mengenai penggunaan ISO/IEC 25010:2011 untuk mengevaluasi sistem informasi berbasi *web framework laravel* yang diimplementasikan pada sebuah sistem *electronic government* di bidang kesehatan yaitu sistem EJKBM (Rahayuda 2017). Pada penelitian ini juga dibahas mengenai penggunaan *white box* dan *black box testing* untuk pengujian sistem (Alfisahrin 2012) (Purnomo 2013).

Pengujian kotak putih

Pengujian kotak putih atau *white box testing* merupakan metode perancangan *test case* yang menggunakan struktur kontrol dari perancangan prosedural untuk mendapatkan *test case.* Dengan menggunakan metode *white box*, analis sistem akan dapat memperoleh *test case* yang menjamin seluruh *independent path* di dalam modul yang dikerjakan sekurangnya sekali, mengerjakan seluruh keputusan logika, mengerjakan seluruh *loop* yang sesuai dengan batasannya dan mengerjakan seluruh struktur data internal yang menjamin validitas (Kruniawati and Widowati 2015) (Mansour and Houri 2017).

Pengujian kotak hitam

Pengujian kotak hitam atau *black box testing* merupakan pengujian sistem yang berfokus pada spesifikasi fungsional dari perangkat lunak. Tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. *Black box testing* bukanlah solusi alternatif dari *white box testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *white box testing* (Mustaqbal, Firdaus, and Rahmadi 2015). *Black box testing* cenderung untuk menemukan hal seperti fungsi yang tidak benar atau tidak ada, kesalahan antarmuka *(interface errors),* kesalahan pada struktur data dan akses basis data, kesalahan performansi *(performance errors)* dan kesalahan inisialisasi dan terminasi. Saat ini terdapat banyak metode atau teknik untuk melaksanakan *black box testing*, seperti, *equivalence partitioning, boundary value analysis* atau *limit testing, comparison testing, sample testing, robustness testing, behavior testing, requirement testing, performance testing*, uji ketahanan *(endurance testing)* dan uji sebab akibat *(cause-effect relationship testing)* (Wahyunningrum and Januarita 2015).

ISO/IEC 25010:2011

Standar ISO/IEC 25010:2011 pertama kali diperkenalkan pada tahun 1991 melalui pertanyaan tentang definisi kualitas perangkat lunak. Dokumen halaman-13 yang asli didesain sebagai fondasi lebih jauh, lebih detail, dan memiliki standard yang dapat diolah. Dokumen standard ISO/IEC 25010:2011 sangat panjang. Hal ini dikarenakan orang memiliki motivasi berbeda yang memungkinkan untuk tertarik pada kualitas perangkat lunak (Maliki and Wiharja 2014).

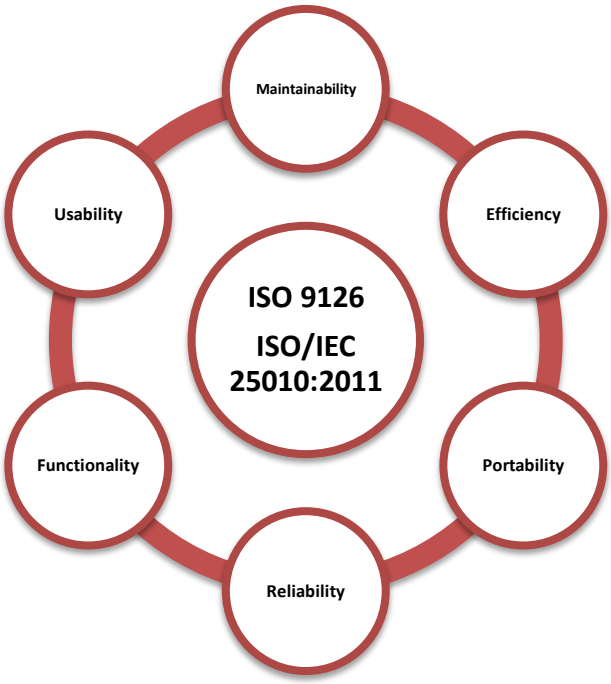
ISO/IEC 25010:2011 telah membagi dokumen menjadi tiga bagian kebutuhan. Disamping ukuran bagian dokumentasi, ISO/IEC 25010:2011 tidak hanya mendefinisikan atribut kualitas perangkat lunak. Standard ISO 14598 memisahkan prosedur yang seharusnya dibawa saat menaksir derajat produk perangkat lunak untuk menyesuaikan diri pada karakteristik kualitas ISO/IEC 25010:2011 yang dipilih. Hal ini mungkin saja tidak diperlukan, tetapi disetujuinya ISO 14598 dapat digunakan

untuk menyelesaikan penilaian dalam membedakan bagian karakteristik kualitas pada ISO/IEC 25010:2011 yang dibutuhkan. Perbedaan antara atribut kualitas internal dan eksternal telah dicatat, ISO/IEC 25010:2011 juga memperkenalkan tipe kualitas *(quality in use)* dimana mengikuti elemen yang telah diketahui (Maliki dan Wiharja, 2014). ISO/IEC 25010:2011 mengidentifikasi enam karakteristik kualitas perangkat lunak utama yaitu:

a. *Functionality* merupakan kemampuan menutupi fungsi produk perangkat lunak yang menyediakan kepuasan kebutuhan *user*.
b. *Reliability* merupakan kemampuan perangkat lunak untuk perawatan dengan level performansi.
c. *Usability* merupakan kemampuan yang berhubungan dengan penggunaan perangkat lunak.
d. *Efficiency* merupakan kemampuan yang berhubungan dengan sumber daya fisik yang digunakan ketika perangkat lunak dijalankan.
e. *Maintainability* merupakan kemampuan yang dibutuhkan untuk membuat perubahan perangkat lunak
f. *Portability* merupakan kemampuan yang berhubungan dengan kemampuan perangkat lunak yang dikirim ke lingkungan berbeda.

METODE

Metode evaluasi dilakukan melalui 2 alur yaitu pengujian sistem dan evaluasi system. Pengujian sistem dilakukan menggunakan metode *white box dan black box testing*, sedangkan evaluasi dilakukan menggunakan ISO/IEC-25010:2011 (Atoum, Bong, and Kulathuramaiyer 2014). ISO/IEC 25010:2011 merupakan standar internasional yang diterbitkan oleh ISO untuk evaluasi kualitas perangkat lunak dan merupakan pengembangan dari ISO 9001. Standar ini dibagi menjadi empat bagian. Tiap bagian menjelaskan mengenai model kualitas, matriks eksternal, matriks internal, dan matriks kualitas yang digunakan. Ada enam ukuran kualitas yang ditetapkan oleh ISO/IEC 25010:2011, yaitu fungsionalitas *(functionality)*, kehandalan *(reliability)*, kegunaan *(usability)*, efisiensi *(efficiency),* portabilitas *(portability),* serta pemeliharaan *(maintainability).* Keenam ukuran kualitas tersebut dapat dilihat pada gambar 1.

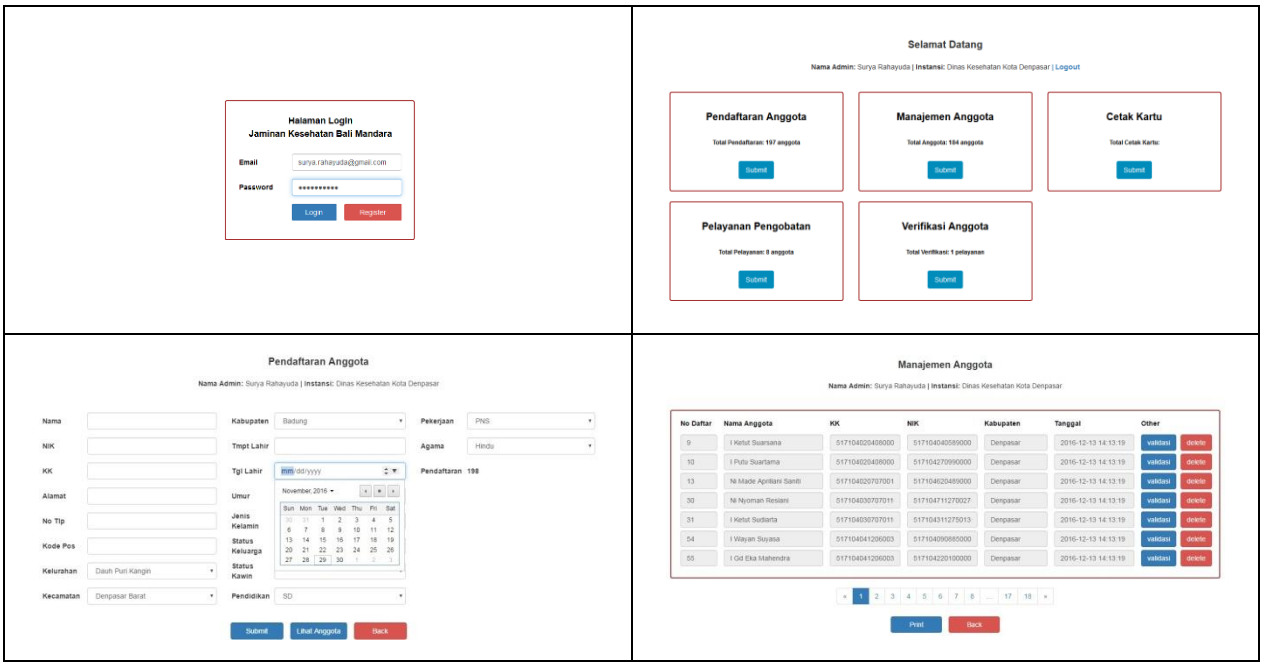


Gambar 1. ISO/IEC 25010:2011. Sumber: Maliki and Wiharja 2014

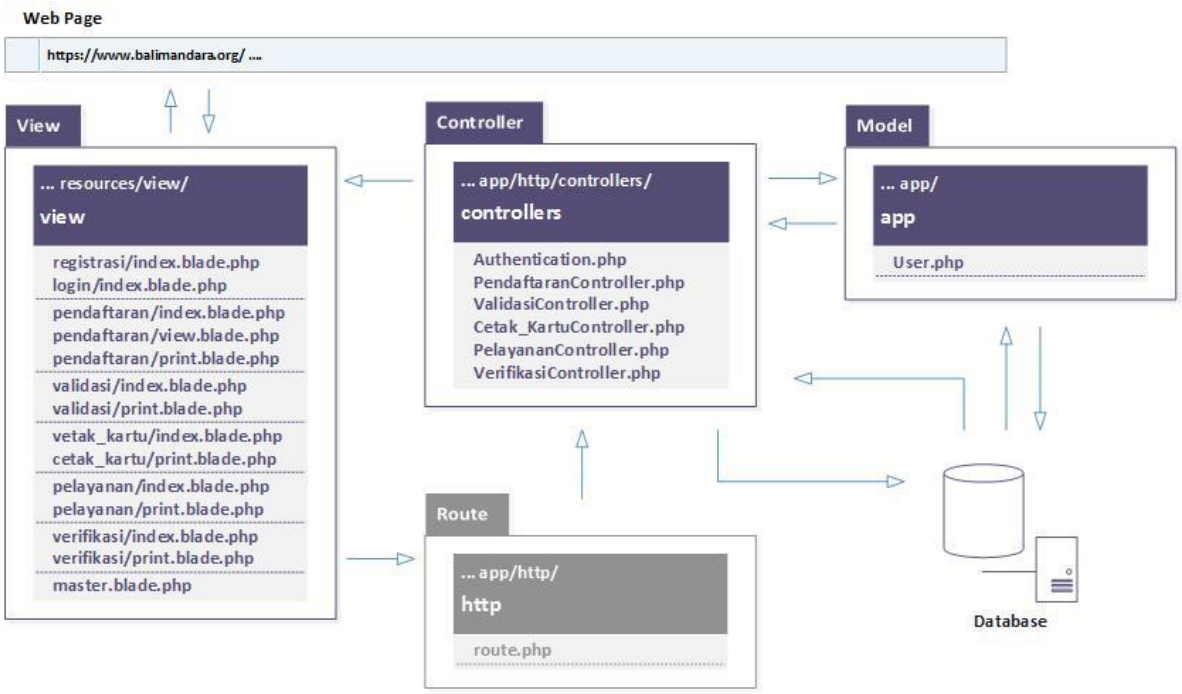
HASIL DAN PEMBAHASAN

Penelitian akan membahas mengenai evaluasi dan hasil evaluasi dari sistem yang dibangun menggunakan metode evaluasi *white box testing, black box testing* dan juga ISO/IEC 25010:2011. *Web* yang akan dievaluasi adalah simulasi *web* yang telah dibangun sebelumnya, yaitu salah satu *web eletronic government* pada bidang kesehatan atau umumnya disebut dengan EJKBM. Simulasi *web* tersebut dibangun

menggunakan *framework laravel*. Tampilan *website* dan semua *file script* yang digunakan dalam sistem dapat dilihat pada gambar 2.



Gambar 2. Beberapa halaman pada sistem web. Sumber: Rahayuda 2017



Gambar 3. Struktur file script program pada MVC sistem web. Sumber: Susanto 2016

Pengujian Sistem

Pengujian juga dilakukan menggunakan metode *white box* dan *black box testing*, *white box testing* merupakan pengujian yang memerlukan pemeriksaan yang detail dan prosedural. Rangkaian logika dari *software* diuji coba dengan menyediakan objek ujicoba yang memerlukan sekumpulan

dari suatu kondisi dan perulangan tertentu. Status program dapat diperiksa dari beberapa titik secara bervariasi untuk menentukan apakah status yang diharapkan atau ditegaskan sesuai dengan status sesungguhnya. Pengujian kotak hitam *(black box test)* khusus didesain untuk mencari kesalahan dengan melakukan ujicoba pada *interface software*. Pegujian kotak hitam *(black box test)* mendemonstrasikan fungsi dari perangkat lunak yang beroperasi, dengan mengecek apakah *input* sudah bisa diterima dengan baik, dan hasil outputnya sesuai dengan apa yang diharapkan, uji coba kotak hitam *(black box test)* melakukan pengecekan pada integritas informasi eksternal, pada dasarnya pengujian kotak hitam *(black box test)* hanya memeriksa hasil *output* yang dihasilkan apakah sudah sesuai dengan apa yang diharapkan dan dinyatakan benar, namun pengujian kotak hitam *(black box test)* tidak mengecek logika dari perangkat lunak.

Pengujian Kotak Putih

Pengujian kotak putih atau *white box testing* dilakukan untuk menemukan kesalahan dalam sistem tersebut. White Box Testing adalah salah satu cara untuk menguji suatu aplikasi atau software dengan cara melihat modul untuk dapat meneliti dan menganalisa kode dari program yang di buat ada yang salah atau tidak. Kalau modul yang telah dan sudah di hasilkan berupa output yang tidak sesuai dengan yang diharapkan maka akan dikompilasi ulang dan di cek kembali kode tersebut hingga sesuai dengan yang diharapkan (Pare 2013). Beberapa proses yang dilakukan dalam pengujian ini, yaitu:

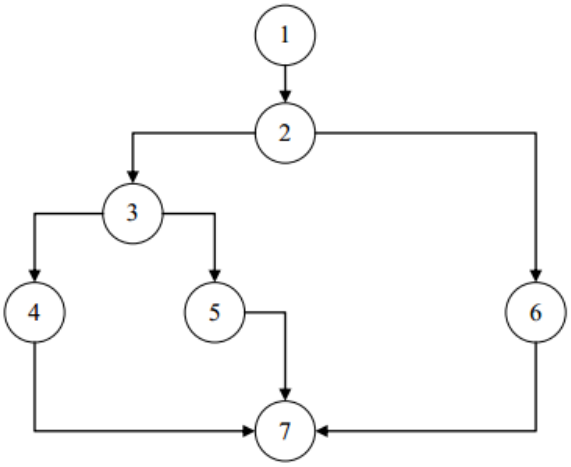
1. Menggunakan perancangan atau kode sebagai sebuah dasar dan tergambar dalam grafik alir yang berfungsi sebagai notasi yang berguna untuk memahami aliran kontrol dan menggambarkan performa.
2. Menentukan kompleksitas siklomatik dari aliran grafik yang dihasilkan, yang berguna untuk memperkirakan modul yang kemungkinan besar akan terbukti salah dan memastikan bahwa semua pernyataan telah dieksekusi minimal sekali. Kompleksitas dihitung dalam salah satu dari tiga cara berikut:

   a. Jumlah daerah grafik alir yang berhubungan dengan kompleksitas siklomat
   b. Komplesitas siklomatik V (G) untuk grafik alir G didefinisikan sebagai
      V (G) = E – N + 2
      Dimana E adalah jumlah *edge* grafik alir dan N adalah jumlah *node* grafik alir
   c. Kompleksitas siklomatik V (G) untuk grafik aliran G juga didefinisikan sebagai V (G) = P + 1
      Dimana P adalah jumlah node predikat yang terdapat dalam grafik alir G
3. Menentukan sebuah basis set dari jalur independen linier, yang berfungsi untuk memperkenalkan setidaknya satu kumpulan pernyataan pemrosesan atau kondisi baru. Berikut pengujian kotak putih pada beberapa *script* program.

```
<?  public function login(LoginRequest $request)                    
    {
        $credentials = $request->only('email', 'password');  (1)
        $credentials =                                       (1)
        [
            'email'     => $request->input('email'),         (1)
            'password'  => $request->input('password')       (1)
        ];

        if(Auth::attempt($credentials))                      (2)
        {
            if (Auth::user()->active == 0)                   (3)
            {
                Auth::logout();                              (4)
                return redirect()->route('info_aktivasi');   (4)
            }
            else                                             (5)
            {
                return redirect()->route('menu');            (5)
            }
        }
        else                                                 (6)
        {
            return redirect()->route('info_error');          (6)
        }
    } ?>                                                     (7)
```

Gambar 4. *Login request* PDL *(Program Design Language)* Sumber: Adithya and Priadi 2014



Gambar 5. Grafix alir *login request* PDL Sumber*:* Alfisahrin 2012

Kopleksitas siklomatik V(G)
V(G) = **3** region jumlah daerah grafik alir
V(G) = 8 edge – 7 + 2 = **3**
V(G) = 2 node prdikat + 1 = **3**
Jadi nilai *cyclomatic complexity* untuk *flowgraph* adalah 3. Jalur independen untuk grafik alir adalah:

Path 1 :
1 – 2 – 3 – 4 – 7 (if (2) = true,
if (3) = true)
Path 2 :
1 – 2 – 3 – 5 – 7 (if (2) = true,
if (3) = false)
Path 3 :
1 – 2 – 3 – 6 – 7 (if (2) = false)

Pengujian kotak putih dapat disimpulkan dari kopleksitas siklomatik V(G), V(G) merupakan hasil mutlaknya dimana nilainya harus sama. Dari beberapa perhitungan yang dilakukan didapatkan nilai V(G) yang sama, dapat disimpulkan bahwa *script* dari program yang telah dibuat adalah relevan (Wijayanto 2014).

Pengujian Kotak Hitam

Pengujian dengan menggunakan metode pengujian kotak hitam atau *black box*, adalah suatu pendekatan untuk dapat menguji dalam setiap fungsi pada suatu program agar dapat berjalan dengan benar (Sriwahyuni, 2011). *Black box testing* adalah metode dimana penguji atau tester hanya mengetahui apa yang harus dilakukan suatu *software*. Penguji tidak mengetahui bagaimana *software* tersebut beroperasi (Subiyakto, 2014). Dengan demikian, penguji hanya menerima hasil dari apa yang dimasukkan tanpa mengetahui bagaimana atau mengapa bisa demikian. Beberapa proses yang dilakuan dalam pengujian ini diantaranya sebagai berikut.

1. Fungsi yang tidak benar, baik *input* atau pun *output*, dalam hal ini hanya melihat apakah proses *input* dan *output* sudah sesuai, contohnya jika ada *software* yang menampilkan *form input* data identitas, jika *user* melengkapi *form* maka program akan melakukan proses simpan, namun jika *user* tidak melengkapi *form* program tidak boleh melakukan proses simpan, jika perangkat lunak tidak sesuai misalnya tidak melengkapi *form* namun dapat tersimpan, hal ini perlu untuk diperbaiki.
2. Kesalahan *interface*, dalam hal kesalahan *interface* sering terjadi pada *software* yang tidak diuji coba dengan baik, misalnya tampilan *web* dengan menggunakan *framework*, ada beberapa *framework* yang tidak mendukung dengan beberapa *browser*, hingga tampilan *interface* kurang maksimal saat *user* memakai *browser* yang tidak mendukung *framework* yang kamu gunakan.
3. Kesalahan dalam struktur data atau akses basis data, yang sering menjadi kendala, karena hal ini dapat berdampak pada akses *web* yang menjadi lambat.
4. Prilaku atau kinerja kesalahan yang ada pada perangkat lunak.
5. Inisialisasi dan penghentian kesalahan pada perangkat lunak.

Tabel 1. Pengujian kotak hitam *(black box testing)*

No	Rancangan Proses	Hasil yang diharapkan	Hasil	Ket.
1	Mengisi *form login* dan klik *login* button	Masuk ke halaman menu utama.	Sesuai	Jika *input* benar
2	Mengisi *form* registrasi dan klik *button* registrasi	Masuk ke halaman *login* dan dapat mengirimkan *email* melalui gmail.	Sesuai	Jika *input* benar
3	Aktivasi a*ccount*	Masuk kehalaman notifikasi aktivasi berhasil.	Sesuai	Jika halaman notifikasi dpat diakses
4	Akses halaman menu	Menampilkan semua menu yang ada. Menampilkan nama dan instansi admin. Menampilkan jumlah data yang telah diproses pada tiap aplikasi	Sesuai	Jika semua nilai ditampilkan
5	*Input* data pendaftaran	*Form* isian dapat digunakan. Data dapat disimpan pada saat *submit*	Sesuai	Jika *input* benar
6	Validasi peserta	Daftar anggota dapat ditampilkan Validasi dapat diproses	Sesuai	Jika muncul tulisan validasi berhasil
7	Update data pendaftaran	Tombol *update* pada halaman administrasi dapat diakses Data yang diinputkan dapat ter-*update*	Sesuai	Jika muncul tulisan *update* berhasil
8	Delete data pendaftaran	Tombol *delete* dapat diakses dan data dipilih dapat terhapus	Sesuai	Jika data yang dipilih dapat terhapus
9	*Search* dan input anggota pada pelayanan	No anggota yang diinputkan pada menu *search* dapat dicari Data pelayanan dapat ditambahkan pada anggota tersebut	Sesuai	Jika no anggota dapat dicari dan data pelayanan dapat diinputkan
10	*Print out* data pelayanan anggota	Data pelayanan dapat ditampilkan dan dapat di *print* dalam bentuk file pdf	Sesuai	Jika data pelayanan dapat di print
11	Verifikasi anggota yang telah terlayani	Tombol verifikasi dapat diakses pada halaman verifikasi anggota Verifikasi anggota dapat diproses dan data keuangan dapat teratat dengan baik	Sesuai	Jika verifikasi anggota berhasil
12	*Print out* laporan pendaftaran, pelayanan dan keuangan	Tombol *print out* pendaftaran, playanan dan keuangan dapat diakses Laporan pendaftaran dapat di *print* dalam bentuk *file pdf* Hasil *print out* sesuai dengan format yang digunakan Hasil *print out* dapat dibagi menjadi beberapa halaman dan terdapat no halaman pada tiap halaman	Sesuai	Jika laporan pendaftaran, pelayanan dan keuangan dapat di *print*

Dari pengujian menggunakan metode *black box testing* didapatkan hasil bahwa beberapa proses yang terdapat pada sistem sesuai dengan hasil yang diharapkan.

Evaluasi ISO/IEC 25010:2011

Evaluasi sistem dilakukan menggunakan ISO/IEC 25010:2011 (Esaki 2013). ISO/IEC 25010:2011 merupakan standar internasional yang diterbitkan oleh ISO untuk evaluasi kualitas perangkat lunak dan merupakan pengembangan dari ISO 9001 (Miguel, Mauricio, and Rodríguez 2014).

Fungsionalitas

Fungsionalitas atau *functionality* merupakan satu set atribut yang bergantung pada keberadaan seperangkat fungsi dan sifat yang ditentukannya (Birla and Johansson 2017). Fungsi adalah fungsi yang memenuhi kebutuhan tersirat atau tersirat. Seperti, kesesuaian, ketepatan, interoperabilitas, keamanan dan kepatuhan fungsi (Acharya and Sinha 2013). Evaluasi mengenai fungsionalitas dilakukan dengan mengevaluasi tiap fungsi atau kegunaan yang wajib atau harusnya dimuliki oleh sistem tersebut. Selain mengenai fungsi, sistem juga akan diuji mengenai keamanannya. Beberapa evaluasi mengenai fungsionalitas sistem akan ditampilkan dalam bentuk tabel.

Tabel 2. Evaluasi fungsionalitas berdasarkan ISO/IEC 25010:2011

Fungsi	Ketepatan	Keamanan
Autentication	100	90
Input Data	90	80
Update Data	90	80
Delete Data	90	80
Print Lap.	85	80
Print Kartu	85	80
Validasi	90	80
Verifikasi	90	80

Berdasarkan semua fungsi yang dievaluasi didapatkan bahwa hasil evaluasi ketepatan atau kesesuaian dari sistem tersebut mendapatkan nilai 90 %. Dan berdasarkan evaluasi mengenai keamanan, sistem mendapatkan nilai 81,25 %.

Keandalan

Keandalan atau *reliability* merupakan satu set atribut yang bergantung pada kemampuan perangkat lunak untuk mempertahankan tingkat kinerjanya dalam kondisi yang dinyatakan untuk jangka waktu tertentu. Seperti toleransi, kematangan, kesalahan, pemulihan, dan kepatuhan keandalan (Pinciroli 2016). Beberapa evaluasi mengenai keandalan system sebagai berikut.

Tabel 3. Evaluasi kehandalan berdasarkan ISO/IEC 25010:2011

Kehandalan Sistem	Bulan 1				Bulan2			
	1	2	3	4	1	2	3	4
Autentifikasi	√	√	√	√	√	√		√
Registrasi	√		√	√	√	√	√	√
Pelayanan	√	√	√		√	√	√	√
Vlidasi	√	√	√	√	√		√	√
Verifikasi	√	√	√		√	√	√	√
Print Lap	√	√			√	√	√	√
Print Kartu		√			√	√	√	√

Kehandalan	Kematangan	Kesalahan
Autentifikasi	87.5	12.5
Registrasi	87.5	12.5
Pelayanan	87.5	12.5
Vlidasi	87.5	12.5
Verifikasi	87.5	12.5
Print Lap	75	25
Print Kartu	62.5	37.5

Evaluasi mengenai keandalan sistem dilakukan dalam dua bulan. Sistem dievaluasi mengenai fungsi apa saja yang ada di dalam system. Fungsi tersebut dievaluasi secara terus menerus setiap hari selama kurun waktu beberapa bulan. Proses evaluasi yang dilakukan selama jangka waktu yang lama dilakukan untuk menguji apakah sistem mampu menghasilkan kinerja yang baik dan konsisten setiap kali sistem itu digunakan dan apakah dapat bertahan dalam jangka waktu yang cukup lama.

Berdasarkan hasil evaluasi yang dilakukan selama beberapa bulan didapatkan bahwa nilai kematangan sistem ialah 82,14%

dan kesalahan yang terjadi pada sistem sebesar 17,86 %.

Penggunaan

Penggunaan atau *usability* merupakan satu set atribut yang sesuai dengan upaya yang dibutuhkan untuk digunakan. Sementara itu, pada penilaian individual, penggunaan tersebut, oleh pengguna dinyatakan atau disiratkan, seperti tentang pengertian, kemampuan belajar, operabilitas, daya Tarik, dan kepatuhan penggunaan (Alhuhud and Altamimi 2016). Beberapa evaluasi mengenai penggunaan.

Tabel 4. Evaluasi penggunaan berdasarkan ISO/IEC 25010:2011

Pengguna (User)	*Ussabilty* Sistem	
	Operabilitas	*Interface*
Agus Novan Prasatya	85	90
I Kadek Agus Indra	95	100
I Nyoman Adi Sastra	80	80
Diva Adi Pradana	100	85
Hermawan Wahyu A.	70	100
I Putu Bayu Krisna H.	70	100
M Rickybastian	85	100
I Kadek Arya Sayoga	100	100
I Gusti Agung Rian G.	90	95
I Gede Devid Wahyu P.	95	90
Ian Adiasa Lee	90	85
Dian Wachid Ibrahim	80	80
I Gede Pande D.	70	80
Ni Kadek Eni Rediastuti	70	80
I Gst Pt Bgs Andika P.	85	90
Putra Adi Wasesa	100	90
I Gede Fian Priantama	100	90
I Putu Sukma Raharja	100	90
Gede Surya Rian H.	85	85
I Gede Ivan Pratama	85	85
I Gede Kusuma Wijaya	80	80
I.B. Desta Kresna K.	100	90
I Wayan Gede Juli S.	100	85
Raymond Kusuma	85	85
Bayu Sastra Geni	90	85
Muammar Qadapi	85	85
Fitria Risci Sugesta	70	85
Wayan Eka Marsa Yella	85	100
I Made Panji W.	95	95
AA Gd Wahyu M.	100	100
I Putu Gede Suarjaya	80	85
I Made Dwija Abinawa	90	85
Naufal Satria Ananta	90	100
Satya Haprabu	80	90
A.A. Indah Puspita K.	85	90
Nicolau Carceres C.	100	100
Lukman Hakim	100	100
Ni Putu Aris Novita D.	80	100
Moh Farih Dzaky	75	85
I Nyoman Arya A.	70	75
Remigius Bria Mali	100	85
Anna Febriana H Nahak	95	100
Gede Putra Yasa	85	100
Panggar Hinggiranja	85	85
Hariati	85	90
Hafidh Bahruddin	80	95
Paskalis Galus	80	95
Simplexius Bria	85	95
Guntur Dwi Anggara	95	95
I Made Palguna Wijaya	80	95
Nevi Syiarningrum	80	85
Ni Made Widyari	80	80
Nicholas Abu Sofyan	95	80
Aris Alnindhom	95	90
I Putu Marta Adi Putra	70	90
I Gede Rifan Nadyarta	70	85
I Putu Ari Wirajaya	80	100
Ahmada Hasbyallah	70	100
Devi Yoga Indiyarti	70	100
Armiyadi	85	100
Wawan Aji Sadewo	100	100
Sofyan Wafiq Hidayat	80	90
I Kadek Mertajaya	70	95
Rendy Prasetyanto Aji	80	85
Ni Luh Putu Yasih S.	95	85
Ni Putu Inten Marpina	100	100
Kd Diky Wiradyatmika	85	100
Septiano Dewanai	80	100
I Putu Agus Yuliartawan	85	90
I Made Putra M.	100	95
I Made Yoga Aditya	90	90
Ni Luh Rima Parahita	85	95
Gusti Ayu Putu Fajar G	100	90
I Gusti Ayu Putu Utari	100	100
Ashry Sundari	100	100
I Kadek Reza Priyatna	90	90

Evaluasi mengenai penggunaan sistem dilakukan dengan cara melakukan survei ke beberapa orang. Survei dilakukan dengan memberikan *link website*, dan *user.* Selanjutnya pengguna akan mengakses *web* tersebut dan menggunakan beberapa fitur yang ada. Setelah menggunakan sistem tersebut, selanjutnya *user* memberikan pendapat mengenai operabilitas dan *interface* sistem. Berdasarkan hasil survei yang dilakukan ke beberapa orang, didapatkan hasil berupa nilai operabilitas sebesar 86,58 % dan *interface* sebesar 91,38%.

Efisiensi

Efisiensi atau *efficiency* merupakan satu set atribut yang berpengaruh pada hubungan antara tingkat kinerja perangkat lunak dan jumlah sumber daya yang digunakan, dalam kondisi yang dinyatakan. Misalnya, perilaku waktu, pemanfaatan sumber daya, dan kepatuhan efisiensi (Pérez et al. 2017). Beberapa evaluasi mengenai efisiensi.

Tabel 5. Evaluasi efisiensi berdasarkan ISO/IEC 25010:2011

Fitur Sistem	Efisiensi Waktu	
	Akses	Proses
Autentication	48 detik	300 detik
Input Data	30 detik	480 detik
Update Data	24 detik	360 detik
Delete Data	12 detik	120 detik
Print Lap.	27 detik	180 detik
Print Kartu	21 detik	240 detik
Validasi	15 detik	120 detik
Verifikasi	18 detik	120 detik

Berdasarkan evaluasi yang dilakukan didapatkan bahwa rata-rata waktu yang digunakan untuk mngakses tiap halaman atau fitur yang terdapat pada sistem berkisar antara 27,86 detik. Dan, waktu yang diperlukan untuk menjalankan tiap proses pada fitur tersebut berkisar antara 274,29 detik atau 4,57 menit. Dari waktu yang diperoleh tersebut dapat disimpulkan bahwa tidak terlalu banyak waktu yang diperlukan untuk mengakses maupun memproses tiap fitur yang terdapat pada sistem.

Pemeliharaan

Pemeliharaan atau *maintainability* merupakan satu set atribut yang sesuai dengan upaya yang diperlukan untuk melakukan modifikasi tertentu. Misalnya, analisis, perubahan, stabilitas, testabilitas, dan kepatuhan pemeliharaan.

Evaluasi mengenai *maintainability* dilakukan pada saat yang bersamaan dengan evaluasi mengenai kehandalan system yang dilakukan selama beberapa bulan. Pada saat evaluasi tersebut ada beberapa saat dimana sistem tidak dapat bekerja dan perlu dilakukan perbaikan (Ronchieri, Pia, and Canaparo 2017).

Dari evaluasi yang dilakukan dalam beberapa kasus terdapat kerusakan pada sistem dan dapat ditangani dengan melakukan pemeliharaan pada sistem (Ronchieri, Pia, and Canaparo 2017). Tidak terlalu banyak kerusakan yang terjadi dan tidak ada kerusakan yang sampai mengharuskan sistem tersebut diganti atau dibuat ulang. Namun dalam suatu kondisi, terkadang ada perbaikan sistem yang harus menghentikan kegiatan sistem selama beberapa waktu. Beberapa kerusakan terjadi pada sistem.

Tabel 6. Evaluasi pemeliharaan berdasarkan ISO/IEC 25010:2011

Kerusakan Sistem	Pemeliharaan		
	kerusakan	perbaikan	penghentian
Input	5	5	0
Data	3	2	1
Print	15	10	5
Update	2	2	0
Validasi	1	1	0
Verifikasi	1	1	0

Berdasarkan evaluasi yang dilakukan didapatkan bahwa pada beberapa kerusakan

sistem yang dapat diperbaiki atau ditangani secara langsung sebesar 77,78% dan perbaikan sistem yang memerlukan penghentian sementara sebesar 28,57 %.

Portabilitas

Portabilitas atau *portability* merupakan satu set atribut yang bergantung pada kemampuan perangkat lunak untuk ditransfer dari satu lingkungan ke lingkungan lainnya (Acharya and Sinha 2013). Kemampuan itu ialah, kemampuan beradaptasi, kemampuan instalasi, hidup berdampingan, penempatan dan kepatuhan portabilitas.

Evaluasi mengenai *portability* dilakukan berdasarkan *browser*, ukuran layar dan resolusi layar. Pengujian ini dilakukan karena umumnya *website* yang dibangun akan mengalami kerusakan atau kesalahan dalam hal tampilan ukuran. Di beberapa *browser* akan ada beberapa fungsi dari *web* yang tidak dapat berjalan dengan baik, begitu juga mengenai permasalahan tampilan. *Web* akan mengalami kerusakan karena ukuran *device* atau layar yang berbeda, misalkan saja aka terdapat beberapa tampilan *website* yang akan terpotong jika diakses dari layar yang berukuran lebih kecil, atau kemungkinan juga akan terdapat kerusakan desain. Dengan menggunakan *framework,* kekurangan tersebut akan dapat diatasi.

Untuk membuktikan hal tersebut, akan dilakukan pengujian sistem. Evaluasi mengenai portabilitas berdasarkan *browser*, ukuran layar dan resolusi layar telah dilakukan pada penelitian sebelumnya. Karena itu pada penelitian kali ini, akan ditampilkan mengenai hasil yang telah didapatkan pada penelitian sebelumnya.

a. Evaluasi pada *browser*

Browser yang digunakan untuk evaluasi adalah *mozilla, chrome, safari* dan *opera*. Sistem diakses pada beberapa *browser* tersebut, kemudian dilakukan semua proses yang ada pada sistem.

Dari evaluasi yang dilakukan didapatkan bahwa semua halaman dan semua proses yang ada pada sistem dan diakses pada semua *browser* tersebut berjalan dengan lancar. Tidak ada kendala yang besar, penggunaan *framework* sangat membantu dalam hal ini. Keserasian desain terlihat sama pada setiap *browser,* hanya sedikit perbedaan yang terlihat. Pada *mozilla* dan *chrome* tampilan *web* optimal. Namun, pada *safari* dan *opera* terdapat sedikit perbedaan dalam tampilan desain, tetapi tidak terlalu signifikan dan tidak menggantu akses dan proses sistem.

b. Evaluasi pada beberapa *device* atau ukuran layar

Selanjutnya, evaluasi juga dilakukan pada beberapa *device* dan juga dilakukan untuk mengevaluasi mengenai hasil dari tampilan *web* jika diakses pada ukuran layar yang berbeda.

Sistem diakses pada beberapa *device,* seperti komputer *desktop* (18,5 inchi), laptop (14 inchi), *tab* (8,4 inchi) dan *handphone* (4 inchi). Dari evaluasi yang dilakukan didapatkan hasil bahwa sistem dapat berfungsi dengan baik, dan tidak ada tampilan web yang terpotong atau hancur ketika diakses pada berbagai *device* dengan ukuran layar yang berbeda

c. Evaluasi pada beberapa resolusi layar

Pengujian yang terakhir ialah pengujian yang dilakukan pada beberapa resolusi layar, karena pada beberapa situs akan mengalami perubahan jika resolusi layar pada komputer dirubah, dan terkadang tampilan situs tersebut tidak dapat tampil dengan baik atau dapat

dikatakan tampilannya hancur, karena tulisan atau ukuran gambar yang terlalu kecil dan sebaliknya. Gambar, tulisan, atau tabel tersebut akan terlihat kacau sehingga membuat situs atau *web* terlihat hancur. Pada penelitian ini evaluasi dilakukan pada beberapa ukuran resolusi layar seperti, 1366x768, 1280x1024, 1024x768 dan 800x600.

Dari evaluasi yang dilakukan pada beberapa resolusi layar tersebut, sistem tetap dapat berfungsi dengan baik dan tampilan sistem tidak menjadi kacau atau hancur. Hanya saja pada ukuran layar 800x600 tampilan menjadi kurang menarik, karena ukuran *font* dan gambar yang terlalu besar dan layar yang beberbentuk agak persegi.

PENUTUP

Dari beberapa perhitungan yang dilakukan pada *white box testing* didapatkan nilai *cyclomatyc complexity* V(G) yang sama, dapat disimpulkan bahwa *script* dari program yang telah dibuat adalah relevan.

Dari pengujian menggunakan metode *black box testing* didapatkan hasil bahwa beberapa proses yang terdapat pada sistem sesuai dengan hasil yang diharapkan.

Berdasarkan semua fungsi yang dievaluasi didapatkan bahwa hasil evaluasi ketepatan atau kesesuaian dari sistem tersebut mendapatkan nilai 90 %. Dan berdasarkan evaluasi mengenai keamanan, sistem didapatkan nilai 81,25%. Dari hasil tersebut dapat disimpulkan evaluasi berdasarkan fungsionalitas dinyatakan sistem memiliki tingakat fungsionalitas yang cukup baik.

Berdasarkan hasil evaluasi yang dilakukan selama beberapa bulan didapatkan bahwa nilai kematangan sistem adalah 82,14 %. Dan kesalahan yang terjadi pada sistem sebesar 17,86 %. Dari hasil tersebut dapat disimpulkan bahwa sistem memiliki tingkat keandalan yang cukup baik.

Berdasarkan hasil survei yang dilakukan ke beberapa orang didapatkan hasil berupa nilai operabilitas sebesar 86,58 % dan interface sebesar 91,38 %. Dari nilai tersebut dinyatakan bahwa sistem yang dibangun memiliki tingkat penggunaan atau usabilitas yang cukup baik.

Berdasarkan evaluasi yang dilakukan didapatkan bahwa rata-rata waktu yang digunakan untuk mengakses tiap halaman atau fitur yang terdapat pada sistem berkisar antara 27,86 detik. Dan waktu yang diperlukan untuk menjalankan tiap proses pada fitur tersebut berkisar antara 274,29 detik atau 4,57 menit. Dari waktu yang diperoleh tersebut dapat disimpulkan bahwa tidak terlalu banyak waktu yang diperlukan untuk mengakses maupun memproses tiap fitur yang terdapat pada sistem. Dapat dikatakan bahwa sistem memiliki tingkat efisiensi yang cukup baik.

Berdasarkan evaluasi yang dilakukan didapatkan bahwa pada beberapa kerusakan sistem yang dapat diperbaiki atau ditangani secara langsung sebesar 77,78% dan perbaikan sistem yang memerlukan penghentian sementara sistem sebesar 28,57 %. Dari hasil tersebut dapat disimpulkan bahwa tidak terlalu banyak krusakan yang dialami pada sistem, dari beberapa kerusakan tresebut sebagian besar dapat ditangani dan hanya sedikit yang memerlukan proses penghentian system sementara. Dapat dikatakan system memiliki tingkat pemeliharaan yang baik.

Dari hasil evaluasi yang dilakukan berdasarkan probabilitas didapatkan bahwa sistem informasi yang dibangun menggunakan *framework* dapat berjalan dengan baik pada tiap *browser* seperti *mozilla*, *chrome*, *safari* dan *opera*. *Framework* dapat berjalan dengan baik pada tiap *device* dengan berbagai ukuran layar seperti *desktop*, *laptop*, *tab* dan *handphone. Framework* dapat berjalan dengan baik pada tiap resolusi layar seperti 1366x768, 1280x1024, 1024x768 dan

800x600

Dari hasil penelitian ini peneliti menyarankan untuk melakukan perbaikan desain sistem agar tampilan *web* dapat lebih baik. Selain itu juga perlu perbaikan kualitas dan struktur basis data untuk meminimalisir kesalahan dan kerusakan data. Perlu juga untuk menambahkan metode pengujian seperti ISO atau *quality assurance* lainnya untuk mengetahui kualitas *software*. Peneliti juga menyarankan untuk menggunakan framework untuk menghitung tingkat kematangan perangkat lunak seperti COBIT dan lainnya.

DAFTAR PUSTAKA

Jurnal Cetak:

Rahayuda, Surya. "Implementasi Teknologi Informasi Untuk Mengembangkan E-Government Menggunakan Framework Laravel." In *Seminar Nasional Teknologi Informasi Dan Multimedia 2017*, edited by Bayu Setiaji, Hastari Utama, Agus Fakhurohman, Hartatik, and Bety Wulansari, 2.4-7. Yogyakarta: Panitia Semnas Teknomedia STMIK AMIKOM Yogyakarta, (2017).

Jurnal Online:

Acharya, Anal, and Devadatta Sinha. "Assessing the Quality of M-Learning Systems Using ISO/IEC 25010." *International Journal of Advanced Computer Research* 3 (12), (2013). Diakses 22 Juni 2017.

Adithya, Eko, and Arum Priadi. "Perancangan Dan Pembuatan Sistem Informasi Persetujuan Perbaikan Dan Pergantian Alat Komputer Berbasis Web (Studi Kasus Pada PT. Lautan Teduh Interniaga)." *E-Journal Universitas Lampung*, (2014). Diakses 24 Juni 2017.

Alfisahrin, Sa'diyah. "Pendekatan White Box Testing Untuk Menentukan Kualitas Perangkat Lunak Dengan Menggunakan Bahasa Pemrograman C++." *Paradigma* XIV (1), (2012): 69–78. Diakses 28 Juni 2017.

Alhuhud, Ghada, and Wejdan Altamimi. "Quality Evaluation of Mobile Game: Miftah Alfasaha." *Hindawi 8*, (2016). Diakses 22 Juni 2017.

Atoum, Issa, Chih Bong, and Narayanan Kulathuramaiyer. "Towards Resolving Software Quality-in-Use Measurement Challenges." *International Journal of Business and Sociaty*, (2014). Diakses 5 Juli 2017.

Birla, Sushil, and Mika Johansson. "Quality Requirements for Software-Dependent Safety-Critical Systems History, Current Status , and Future Needs." *NRC Research Press*, no. 1 (2017). Diakses 24 Juni 2017.

Esaki, Kazuhiro. "Verification of Quality Requirement Method Based on the SQuaRE System Quality Model." *American Journal of Operations Research* 3, (2013). Diakses 22 Juni 2017: 70–79.

Kruniawati, Suri, and Sri Widowati. "Implementasi Metode Cause Effect Graphing (CEG) Dalam Pengujian Requirement Perangkat Lunak (Studi Kasus: Aplikasi G-College)." *E-Proceeding of Engineering* 2 (2), (2015): 6475–80. Diakses 28 Juni 2017.

Maliki, Reza, and Kemas Wiharja. "Implementasi Iso 25010:2010 Untuk Evaluasi Kualitas Perangkat Lunak (Studi Kasus : I-Gracias Universitas Telkom)." Universitas Telkom, (2014). Diakses 28 Juni 2017.

Mansour, Nashat, and Manal Houri. "White Box Testing Of Web Applications." *Journal of Systm and Software*, (2017):1–9. Diakses 5 Juli 2017.

Miguel, José P, David Mauricio, and Glen Rodríguez. "A Review of Software Quality Models for the Evaluation of Software Products." *International Journal of Software Engineering and Applications* 5 (6), (2014): 31–53. Diakses 5 Juli 2017.

Mustaqbal, M Sidi, Roeri Firdaus, and Hendra Rahmadi. "Pengujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis (Studi Kasus : Aplikasi Prediksi Kelulusan SNMPTN)." *Jurnal Ilmiah Teknologi Informasi Terapan* 1 (3), (2015): 31–36. Diakses 5 Juli 2017.

Pare, Selfina. "Desain Dan Implementasi E-Commerce Pada Toko As 88 Celluler Merauke." *Jurnal Ilmiah Mustek Anim Ha* 2 (3), (2013): 222–29. Diakses 5 Juli 2017.

Pérez, César, Vicente Montequín, Francisco Fernández, and Joaquín Balsera. "Integrating

Analytic Hierarchy Process (AHP) and Balanced Scorecard (BSC) Framework for Sustainable Business in a Software Factory in the Financial Sector." *MDPI* (2017). Diakses 5 Juli 2017. doi:10.3390/su9040486.

Pinciroli, Fernando. "Improving Software Applications Quality by Considering the Contribution Relationship among Quality Attributes." *Procedia - Procedia Computer Science* 83 (Antifragile). Elsevier Masson SAS, (2016): 970–75. Diakses 5 Juli 2017. doi:10.1016/j.procs.2016.04.194.

Purnomo, Adi. "Software Testing Aplikasi Website PT. Gramedia Menggunakan Metode Blackbox Pada PT WGS Bandung." *E-Journal Universitas Dianapura*, (2013). Diakses 28 Juni 2017.

Ronchieri, Elisabetta, Maria Pia, and Marco Canaparo. "Geant4 Maintainability Assessed with Respect to Software Engineering References." *arXiv*, (2017). Diakses 22 Juni 2017.

Sriwahyuni, Titi. "Implementasi Perancangan Sistem Informasi Ekspedisi Paket Pada PT. POS Indonesia." *Jurnal Teknologi Informasi Dan Pendidikan* 4 (1), (2011). Diakses 5 Juli 2017.

Subiyakto, A'ang. "Pengembangan Aplikasi Jurnal Elektronik Fakultas Sains Dan Teknologi Berbasiskan Web." *E-Journal Syarif Hidayatullah State Islamic University Jakarta*, (2014): 1–10. Diakses 24 Juni 2017.

Susanto, Moch. "Pengukuran Software Metric Terhadap Implementasi Framework Laravel Pada Pembangunan Aplikasi Berbasis Web Studi Kasus: Jurnal Logic Software Metric Measurement on Laravel Framework Impelementation for Website Application Case Studi : Jurnal Logic." Telkom University, (2016). Diakses 5 Juli 2017.

Wahyunningrum, Tenia, and Dwi Januarita. "Implementasi Dan Pengujian Web E-Commerce Untuk Produk Unggulan Desa." *Jurnal Komputer Terapan* 1 (1), (2015): 57–66. Diakses 5 Juli 2017.

Wijayanto, Ridho. "Perancangan Animasi Interaktif Pembelajaran Bahasa Inggris Untuk Kelas 2 Pada Mi Nurul Falah Ciater." *E-Journal Bina Sarana Informatika* 2 (1), (2014): 1–11. Diakses 22 Juni 2017.