Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 3, No. 9, September 2019, hlm. 9314-9320 http://j-ptiik.ub.ac.id

Pengembangan Aplikasi Penentuan Prioritas Kebutuhan Perangkat Lunak dengan Metode *Value Oriented Prioritization* (VOP)

Hanida Aisha[1], Denny Sagita Rusdianto[2], Agi Putra Kharisma[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]hanidaaisha@gmail.com, [2]denny.sagita@ub.ac.id, [3]agi@ub.ac.id

Abstrak

Keberhasilan perangkat lunak tergantung pada pemilihan kebutuhan yang diprioritaskan dengan baik. Prioritas kebutuhan dianggap sebagai salah satu pendekatan yang paling penting dalam proses rekayasa kebutuhan. Prioritisasi kebutuhan digunakan untuk menentukan pemesanan atau jadwal untuk melaksanakan kebutuhan berdasarkan prioritas atau kepentingannya sehubungan dengan sudut pandang para pemangku kepentingan. Banyak teknik atau metode prioritas kebutuhan yang telah diusulkan oleh para peneliti. Pada penelitian ini menerapkan metode yang diusulkan oleh Jim Azar, Randy K.Smith dan David Cordes untuk membantu memprioritaskan kebutuhan yang satu-satunya menggunakan nilai bisnis inti. Metode *Value Oriented Prioritization* adalah metode yang menekankan pada nilai-nilai bisnis utama dalam proses penentuan prioritas, menggunakan skala ordinal sesuai dengan seberapa pentingnya *business core* tersebut dalam perusahaan. Resiko dari bisnis yang dijalankan perusahaan juga ikut diidentifikasi dan diberikan nilai positif untuk pembobotan *business core*, sedangkan nilai negatif untuk pembobotan resiko. Aplikasi ini telah dikakukan pengujian untuk kebutuhan fungsional dan non-fungsional, kebutuhan fungsional menggunakan *black-box testing* yaitu pengujian validasi dan *white-box testing* yaitu pengujian unit dan integrasi serta kebutuhan non-fungsional menggunakan pengujian *compatibility* dengan bantuan *software sortsite* dengan hasil tidak mendapatkan *critical issue* pada banyak *browser* dan pengujian akurasi sehingga dapat disimpulkan mendapatkan hasil yang *valid*.

Kata kunci: *kebutuhan fungsional, rekayasa kebutuhan, prioritisasi kebutuhan, value oriented prioritization*

Abstract

The success of the software depends on the selection of needs that are prioritized properly. Priority needs are considered as one of the most important approaches in the process of needs engineering. needs prioritization is used to determine the bookings or the schedules to carry out the needs based on their priorities or interests with respect to the stakeholders' point of view. There are various techniques and methods that have been proposed by the researchers. In this study, the researcher applies the priority method proposed by Jim Azar, Randy K. Smith, and David Cordes to prioritized the software needs which only use a core business value. The Value-Oriented method Prioritization is a method that emphasizes the main business value in the process of determining the priority, using an ordinal scale according to how important the business in the company is. the risk of the business carried on by the company is also identified and given a positive value for weighting the business core, while a negative value for weighting the risk. This application has been tested for functional and non functional requirements, functional requirements using black-box testing, namely validation testing, and white box testing, namely unit testing and integration and non functional requirements using compatibility testing with the help of sortsite software with the result of not getting critical issues in many browsers and acuuracy testing, so it can be concluded that it gets a valid results.

Keywords: *functional requirement, requirement engineering, requirement prioritization, value oriented prioritization*

1. PENDAHULUAN

Requirement Engineering merupakan proses terstruktur dari elicitating, defening, negotiating, prioritizing, dan validating kebutuhan dari sistem. Pada proses prioritizing mempunyai kedudukan penting dalam melakukan pengembangan sebuah perangkat lunak. Biasanya, jumlah kebutuhan dari pemangku kepentingan melebihi jumlah fitur yang dapat diimplementasikan dalam waktu dan sumber daya yang tersedia. Sehingga, beberapa fitur yang diminta tidak akan selesai tepat waktu atau dipindahkan ke rilis yang lebih baru. Oleh karena itu, pemangku kepentingan dan tim pengembangan harus memutuskan fungsi apa yang paling penting yang harus diimplementasikan terlebih dahulu (Narendhar dan Anuradha, 2016).

Keberhasilan pengembangan suatu perangkat lunak yang berkualitas tergantung pada pemilihan kandidat kebutuhan yang diprioritaskan berdasarkan pada aspek prioritas utama. Menentukan kebutuhan mana yang harus diimplemntasikan terlebih dahulu merupakan proses pengambilan keputusan yang sangat kompleks (Sher et al., 2014) . Terdapat berbagai cara atau metode yang digunakan untuk membantu menjawab permasalahan prioritas kebutuhan salah satunya adalah metode yang diusulkan oleh Jim Azar, Randy K. Smith dan David Cordes yang menjelaskan satu metode untuk membantu memprioritaskan kebutuhan yaitu Value Oriented Prioritization (VOP). Metode ini menekankan pada nilai-nilai bisnis utama yang menghasilkan kepuasan setiap pemangku kepentingan. VOP membuat matriks prioritas dengan membuat perbandingan menggunakan nilai-nilai dan resiko-resiko bisnis inti. Sehingga dapat memprioritaskan skor untuk setiap kebutuhan sebagai jumlah dari kontribusinya terhadap nilai bisnis dikurangi dengan jumlah resikonya (Azar, Smith dan Cordes, 2007).

Berdasarkan penelitian yang dilakukan oleh Manju Khari dan Nikunj Kumar, peneliti melakukan perbandingan enam metode penentun prioritas kebutuhan software yaitu AHP (Analytic Hierarchy Process), VOP (Value Oriented Prioritization), CV (Cumulative Voting), NAT (Numerical Assigment Technique), BTS (Binary Search Tree), dan PG (Planning Game). Berdasarkan beberapa kriteria yaitu kemudahan dalam penggunaan, waktu, skalabilitas, keakuratan dan jumlah kebutuhan yang akan dibandingkan disimpulkan bahwa metode Value Oriented Prioritization (VOP) adalah metode yang paling baik, mudah digunakan dan memberikan hasil yang akurat meskipun dalam kasus jumlah kebutuhan yang banyak (Khari dan Kumar, 2013).

Sehingga melalui hal tersebut mengambil judul "Pengembangan Aplikasi Penentuan Prioritas Kebutuhan Perangkat Lunak dengan metode Value Oriented Prioritization (VOP). Dengan menerapkan metode yang berasal dari penelitian yang dilakukan oleh Jim Azar, Randy K. Smith dan David Cordes yang berjudul "Value-Oriented Requirements Prioritization in a Small Development Organization". Diharapkan dengan menggunakan metode tersebut dapat membuat software yang nantinnya dapat membantu *system analyst* dalam memprioritaskan kebutuhan.

2. *METODE VALUE ORIENTED PRIORITIZATION* (VOP)

Value Oriented Prioritization (VOP) yaitu metode satu-satunya yang mempertimbangkan nilai bisnis value dalam proses penentuan prioritas. VOP memiliki kerangka kerja yaitu langkah pertama yang dikerjakan dalam metode tersebut yaitu mengidentifikasi nilai bisnis value dalam perusahaan, selanjutnya memberikan prioritas menggunakan skala ordinal sesuai dengan seberapa pentingnya business core tersebut dalam perusahaan. Resiko dari bisnis yang dijalankan perusahaan juga ikut diidentifikasi. Diberikan nilai positif untuk pembobotan business core, sedangkan nilai negatif untuk pembobotan resiko.

Langkah berikutnya adalah mengidentifikasi user requirements, functional requirements, dan non-fuctional requirements. kemudian, dalam setiap requirements dilakukan perbandingan setiap business core dan resiko, dan diberikan bobot. Metode VOP membangun sebuah matriks prioritas yang menggabungkan lima nilai bisnis dan dua tipe resiko seperti pada Gambar 1.

Requirements	Core business values ($V_1 \dots V_n$)					Resiko ($R_1 \dots R_n$)		Score
	Sales $V_1 = 7$	Marketing $V_2 = 6$	Competitive $V_3 = 8$	Strategic $V_4 = 10$	Customer Retention $V_5 = 7$	Technical $R_1 = -8$	Business $R_2 = -5$	
r_1	$W_{i,j}$					$W'_{i,j}$		S
r_2								
...								
r_n								

Gambar 1. Matrks Prioritas VOP

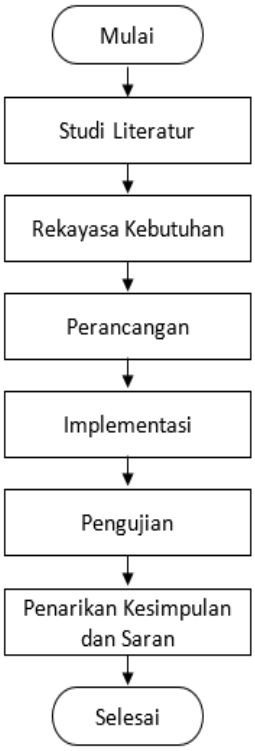
Untuk menghitung nilai prioritas kebutuhan, dilakukan dengan Persamaan (1):

$$\boldsymbol{S = \sum_{i=0}^{n}(Vi \times Wi,j) + \sum_{i=1}^{n}(Rj \times W'i,j)} \quad \textbf{(1)}$$

Keterangan :
S = Score
r_i = kebutuhan ke i
V_i = bobot dari nilai bisnis
R_j= bobot dari resiko j
$W_{i,j}$ = bobot r_i sehubungan dengan nilai bisnis V_j
$W'_{i,j}$ = bobot r_i sehubungan dengan resiko R_j

3. METODOLOGI PENELITIAN

Metodologi yang dipergunakan didalam penelitian kali ini mempergunakan SDLC *waterfall*. Tahapan yang dikerjakan pada metodologi ini yakni studi literatur, analisis kebutuhan, perancangan, implementasi, pengujian, dan penarikan kesimpulan. Diagram alur proses metodologi penelitian kali ini dapat dilihat pada Gambar 2.



Gambar 2. Diagram Alir

Tahapan pertama pada penelitian yang dilakukan yakni studi literatur yang merupakan penjelasan mengenai teori mendasar didalam penelitian bersumber dari buku, jurnal, halaman website, maupun sumber penelitian lain yang terkait dipergunakan sebagai referensi untuk mengadakan pengembangan aplikasi prioritas kebutuhan perangkat lunak dengan menggunakan metode *Value Oriented Prioritization* (VOP).

Selanjutnya rekayasa kebutuhan, yang mendeskripsikan tahapan untuk mewujudkan kebutuhan-kebutuhan yang dibutuhkan oleh pemangku kepentingan didalam mewujudkan suatu aplikasi ketika akan dibangun dan dioperasikan. Rekayasa kebutuhan meliputi elisitasi kebutuhan yang digunakan untuk mengidentifikasi semua kebutuhan aplikasi dan aktor yang berperan didalamnnya yang didapatkan dengan menganalisis penelitian pemrioritasan kebutuhan perangkat lunak yang dilakukan oleh Randy K. Smith dan David Cordes yang berjudul "Value-Oriented Requirements Prioritization in a Small Development Organization". Selanjutnya hasil yang didapatkan dari tahap elisistasi dapat dispesifikasikan dan dimodelkan menjadi diagram *usecase* dan *scenario usecase*.

Tahap ketiga yakni perancangan aplikasi, yang terdiri dari perancangan arsitektur aplikasi dimana akan di buat menggunakan diagram *sequence* maupun diagram *class* yang menggunakan pemodelan UML, perancangan berbasis komponen yang di ambil dari dari sampel algoritma pada setiap *method* kemudian di tuliskan ke dalam bentuk *pseudocode*, perancangan basis data yaitu rancangan tabel *database* menggunakan CDM (*Conceptual Data Model*) dengan memodelkan entitas pada aplikasi hingga menjadi PDM (*Physical Data Model*), Selanjutnya perancangan antarmuka (*interface*) yakni tampilan tata letak *interface* aplikasi yang nantinya di bangun berdasarkan kebutuhan system dan diberi beberapa sampel antarmuka utama berdasarkan level pengguna.

Tahap keempat yakni implementasi untuk mewujudkan konsep yang telah dikerjakan berbentuk kode program dan nantinnya akan menjadi aplikasi yang dapat difungsikan. Implementasi dalam bentuk kode program dilakukan dengan memanfaatkan *framework* CodeIgniter, untuk implementasi pada tampilan antarmuka digunakan *framework* MaterializeCSS, sedangkan implementasi pada struktur data menggunakan MySQL.

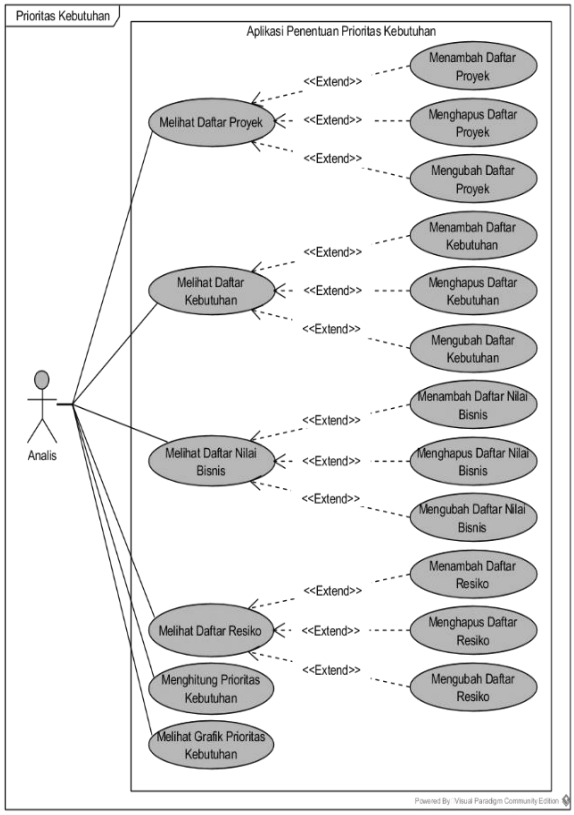
Tahap kelima yakni pengujian tahap implementasi terselesaikan, dilanjutkan pengujian fungsionalitas dan non fungsionalitas aplikasi yang memiliki tujuan menemukan kesalahan pada aplikasi yang telah dibangun berikut ini apakah sudah selaras dengan permintaan kebutuhan yang dijabarkan

sebelumnnya Pada pengujian fungsionalitas mengaplikasikan pengujian unit dan pengujian integrase mempergunakan teknik pengujian *white-box* dan pengujian validasi mempergunakan teknik pengujian *black-box*. Selanjutnya pada pengujian non fungsionalitas mempergunakan pengujian *compatibility* menggunakan bantuan *software Sorsite* dan pengujian akurasi.

Tahapan terkhir ini berisikan rumusan final dari jawaban rumusan masalah pada penelitian yang sudah dijelasakan sebelumnya, kemudian diberikan saran sehingga diharapkan pengembangan yang medatang dapat lebih dikembangkan.

4. REKAYASA KEBUTUHAN

Setelah melakukan analisis didapatkan delapan belas kebutuhan fungsional aplikasi dan satu kebutuhan non-fungsional. Selain daripada itu *actor* yang didapatkan yakni *system analyst* diagram *usecase* aplikasi yang dikembangkan ditunjukkan pada Gambar .



Gambar 3. Diagram *Use Case* Aplikasi

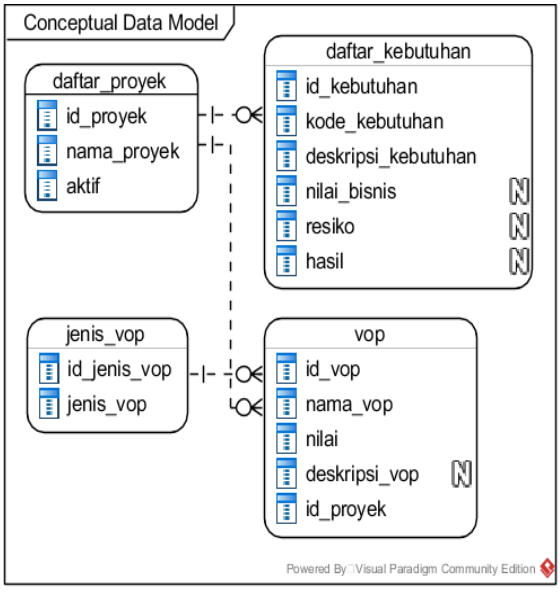
5. PERANCANGAN

Setelah selesai mendefinisikan kebutuhan aplikasi didalam tahapan analisis, maka langkah berikutnya melakukan tahapan perancangan. Perancangan yang dilakukan pada pengembangan aplikasi ini menggunakan perancangan berbasis komponen, perancangan arsitektur aplikasi, perancangan antarmuka (*interface*), dan perancangan bais data.

Perancangan arsitektur aplikasi merupakan perancangan yang dikerjakan dengan menggunakan *sequence* dan *class diagram*. *Sequence diagram* menjelaskan alur rincian pertukaran data antar objek yang ada di dalamaplikasi, sedangan *class diagram* menjelaskan hubungan antar klas penyusun aplikasi.

Pada aplikasi yang dibuat terdiri dari 15 klas. Terdapat 2 klas yang penting karena memakai frameword CI (Code Igniter) yaitu klas CI_controller dan CI_model yang berguna untuk menghubungkan berbagai klas *controller* dan klas *model*. Pada klas *controller* terdapat 5 klas, klas *model* terdapat 5 klas dan klas *views* terdapat 5 klas.

Perancangan data dikerjakan dengan mempergunakan CDM (Conceptual Data Model) dan PDM (Physical Data Model). CDM ini dilakukan dengan membuat model entitas yang berada dalam aplikasi , atribut yang dipunyai oleh entitas, dan relasi yang ada dalam entitas. CDM kemudian digunakan menjadi dasar untuk memodelkan ke bentuk PDM. PDM ini merupakan pemodelan yang lebih lengkap dibandingkan dengan CDM. CDM dari aplikasi ini terdapat pada Gambar 4.



Gambar 4. Model Data Konseptual (CDM)

Perancangan berbasis komponen dilaksanakan untuk mendeskripsikan algoritma

dari proses yang ada dalam komponen – komponen aplikasi . Perancangan ini ini diwakili oleh *pseudocode* dari method tambah_daftar_kebutuhan yang terdapat pada klas c_daftar_kebutuhan, method tambah_nilai_bisnis yang terdapat pada klas c_nilai_bisnis_resiko, dan method hitung_prioritas yang terdapat pada klas c_penentuan_prioritas_keb.

6. IMPLEMENTASI

Implementasi yang dikerjakan penelitian ini terdapat 3 poin, yakni implementasi dalam bentuk kode program, implementasi pada struktur data aplikasi, dan implementasi antarmuka (*interface*). Implementasi dalam bentuk kode program dilakukan dengan merealisasikan *pseudocode* dijadikan dalam kode program memanfaatkan framework CodeIgniter, MySQL, dan juga Bootstrap. Implementasi pada struktur data aplikasi dilakukan dengan merealisasikan PDM (*Physical Data Model*), yang didapatkan pada perancangan basis data dijadikan basis data aplikasi yang lebih lengkap setiap masing-masing atribut diberikan tipe data dan juga atribut. Implementasi basis data aplikasi dapat dilihat pada Gambar 5. Implementasi yang terakhir yakni implementasi antarmuka dari salah satu contoh terdapat pada Gambar 6.

7. PENGUJIAN

Pengujian pada tahahapan ini memiliki tujuan menemukan kesalahan yang ada pada aplikasi yag telah dibangun apakah sudah sesuai dengan permintaan kebutuhan yang sudah di didapatkan sebelumnnya. Dan juga memastikan bahwa aplikasi yang telah dibangun sudah tidak terdapat cacat atau *bug*. Penelitian ini, mengaplikasikan pengujian *white-box* untuk pengujian unit dan pengujian integrasi dan teknik pengujian *black-box* yakni pengujian validasi. Selanjutnya pengujian non fungsionalitas menggunakan pengujian *compatibility* dan juga pengujian akurasi.

Pengujian unit dipergunakan untuk memverivikasi bahwa sepotong kode yang *relative* kecil melakukan apa yang seharusnya Pengujian unit dalam pengembangan aplikasi ini dilakukan pada 3 *sample*, yaitu klas c_daftar_kebutuhan dengan *method* tambah_daftar_kebutuhan, klas c_nilai_bisnis_resiko dengan *method* tambah_nilai_bisnis, dan klas c_nilai_bisnis_resiko dengan *method* tambah_resiko. Setiap kasus uji didapatkan hasil nilai kompleksitas *cyclomatic* sebanyak 2, 3, dan 3 dengan hasil valid pada ketiganya.

Pengujian integrasi dipergunakan untuk memvalidasi satu, dua unit dengan yang lainnya dapat bekerja sama dengan tepat ketika saling terhubung satu sama lain. Pengujian ini dijalankan pada *method* hitung_prioritas yang terdapat pada klas c_penentuan_prioritas_keb yang di dalamnya memanggil *method* vopData dan *method* hitung_prioritas_kebutuhan dari klas m_penentuan_prioritas_keb dan didapatkan hasil nilai kompleksitas *cyclomatic* sebanyak 3 dengan hasil valid.

Pengujian validasi untuk mendeteksi kesalahan yang terjadi pada fungsi – fungsi dari aplikasi yang di kembangkan baik dari segi pembuatan user interface, kinerja sistem, dan kelengkapan fungsionalitas. Terdapat 48 kasus uji dari 19 kebutuhan yang telah terdefinisikan dan dari keseluruhan kasus uji mendapatkan hasil yang valid. Tabel 1 adalah contoh dari salah satu kasus dan hasil uji pada pengujian validasi.

Pengujian *compatibility* pengujian yang menggunakan bantuan software *Sorsite* biasa digunakan untuk meninjau apakah aplikasi yang telah dikembangkan sanggup dijalankan diberbagai jaringan yang beda, dan didapatkan hasil bahwa pengujian yang telah dilakukan sudah teruji valid dikarenakan tidak ada *critical issue* dan dapat dijalankan pada semua *browser*.

Pengujian akurasi dipergunakan didalam memastikan kesesuaiaan perhitungan yang dilakukan manual dengan menghitung menggunakan aplikasi. Dari pengujian akurasi memperoleh hasil yang sama sehingga dapat disimpulkan mendapat hasil yang valid.

Tabel 1. Kasus dan Hasil Uji Menambah Daftar Proyek

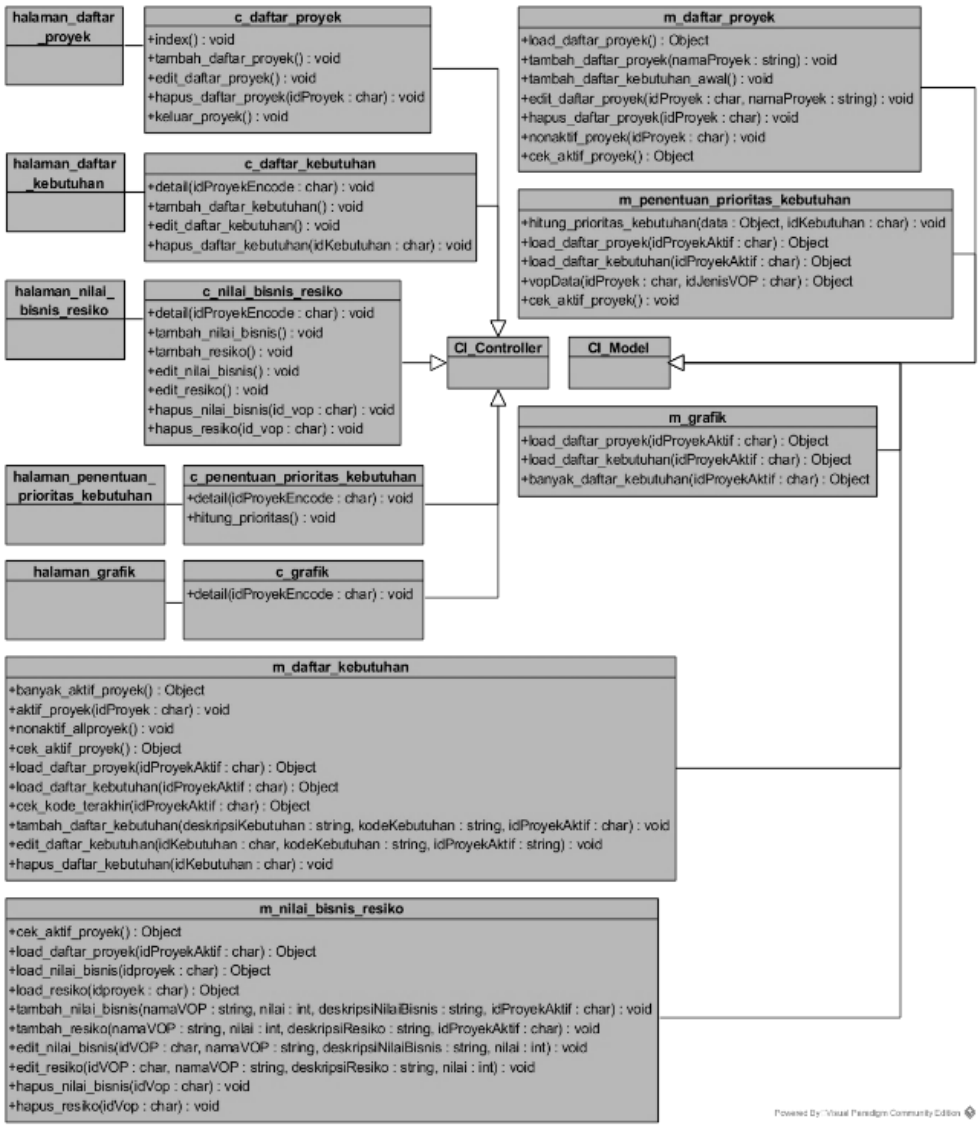
Kode Kebutuhan	PKPL-F-02
Nama Kasus Uji	Menambah daftar proyek kondisi berhasil
Prosedur	1. Mengakses halaman daftar proyek 2. Menekan tombol “Tambah Data” pada table daftar proyek. 3. Mengisi form tambah data proyek dengan rincian : Nama proyek = Proyek A

	4. Menekan tombol "Tambah" pada form tambah daftar proyek
Hasil yang Diharapkan	Data proyek tersimpan dalam database dan ditampilkan pada halaman daftar proyek
Hasil Aktual	Data proyek tersimpan dalam database dan ditampilkan pada halaman daftar proyek
Status	Valid

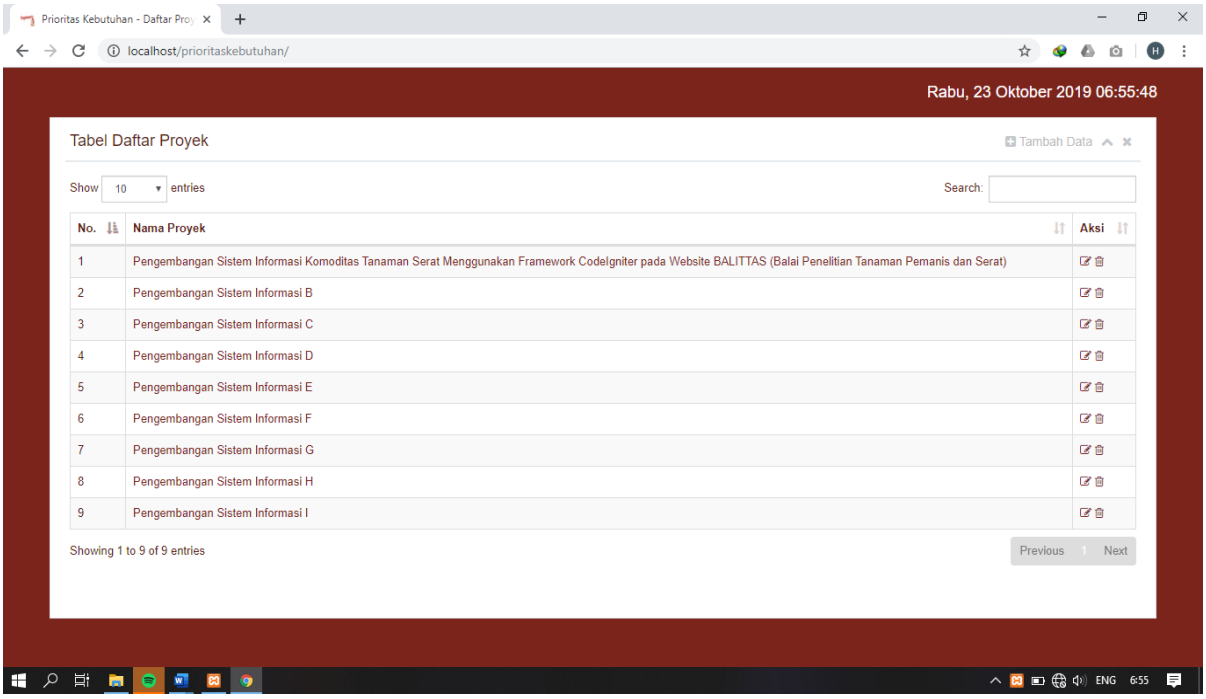
8. KESIMPULAN

Berdasarkan keseluruhan poin-poin yang sebelumnnya telah dijelaskan, terdapat beberapa poin simpulan akhir yakni fase analisis kebutuhan menghasilkan menghasilkan 19 kebutuhan yaitu 18 kebutuhan fungsional dan 2 kebutuhan non-fungsional dan juga satu *actor* yakni *system analyst* yang berperan penuh pada aplikasi.

Fase perancangan arsitektur terdapat 2 macam diagram, yaitu diagram *class* terdiri dari 15 klas, yaitu 5 klas sebagai *controller*, 5 klas sebagai *model*, dan 5 klas sebagai *view* dan diagram *sequence* dibuat dari 3 sampel fungsionalitas aplikasi, yaitu fungsi menambah daftar kebutuhan, menambah daftar nilai bisnis, dan menghitung prioritas kebutuhan. Fase perancangan data menggunakan CDM (*Conseptual Data Model*) dan PDM (*Physical Data Model*). Kedua model data tersebut akan berperan sebagai tolak ukur membangun basis data di dalam aplikasi. Di aspek perancangan komponen, terdapat 3 contoh operasi dari klas-klas yang terdapat pada aplikasi untuk dimodelkan kedalam bentuk *pseudocode*. Pada fase perancangan antarmuka, rancangannya merupakan gambaran implementasi dari halaman antarmuka pada aplikasi.



Gambar 5. Implementasi Data Aplikasi



Gambar 6. Implementasi Antarmuka

Fase implementasi yang digunakan untuk mewujudkan keseluruhan fase yang terjadi didalam bagian perancangan. Sehingga dari perancangan berbasis komponen mendapatkan hasil kode program, struktur basis data aplikasi sebagai bentuk perwujudan perancangan struktur data, dan antarmuka aplikasi sebagai bentuk perwujudan perancangan tampilan antarmuka (*interface*).

Bagian pengujian terdapat pengujian unit di tiga *method* dari klas berbeda dan mendapatkan hasil nilai kompleksitas yang valid pada ketiganya, kemudian terdapat pengujian integrasi untuk bagian yang saling berkorelasi satu dengan lainnya dan mendapat kompleksitas *cyclomatic* berjumlah 3 bernilai valid, dan terdapat pengujian validasi untuk mengecek semua fungsionalitas yang berjumlah 48 kasus uji, pengujian *compatibelity* dan juga pengujian akurasi untuk kebutuhan non-fungsional bernilai valid. Sehingga dari keseluruhan hasil uji dapat didimpulkan bahwa aplikasi dapat dijadikan alat untuk merekomendasikan penentuan prioritas kebutuhan.

Berdasarkan keseluruhan perolehan penelitian, saran yang dapat dijadikan referensi bagi penelitian yang dikerjakan berikutnya yaitu aplikasi prioritas kebutuhan yang dikembangkan sekarang ini memiliki tampilan yang kurang menarik sehingga diharapkan bisa dilakukan perbaikan pada tampilan agar lebih menarik kedepannya.

9. DAFTAR PUSTAKA

Narendhar, M. dan Anuradha, K. (2016) "Different Approaches of Software Requirement Prioritization," *International Journal of Engineering Science Invention*, 5(9), hal. 38–43.

Sher, F. et al. (2014) "Requirements Prioritization Techniques and Different Aspects for Prioritization," (December). doi: 10.1109/MySec.2014.6985985.

Azar, J., Smith, R. K. dan Cordes, D. (2007) "Value-oriented requirements prioritization in a small development organization," IEEE Software, 24(1), hal. 32–37. doi: 10.1109/MS.2007.30.

Khari, M. dan Kumar, N. (2013) "Available Online at www.jgrcs.info COMPARISON OF SIX PRIORITIZATION TECHNIQUES FOR SOFTWARE," Journal of Global Research in Computer Science, 4(1), hal. 38–43.