BAB II

KAJIAN TEORI

A. Deskripsi Teori

1. Beasiswa

Menurut Kamus Besar Bahasa Indonesia, beasiswa merupakan tunjangan yang diberikan kepada pelajar atau mahasiswa sebagai bantuan biaya belajar. Peraturan Pemerintah Nomor 48 tahun 2008 tentang Pendanaan Pendidikan, Bagian Kelima, Pasal 27 ayat (1), menyebutkan bahwa Pemerintah dan pemerintah daerah sesuai kewenangannya memberi bantuan biaya pendidikan atau beasiswa kepada peserta didik yang orang tua atau walinya tidak mampu membiayai pendidikannya. Pasal 27 ayat (2), menyebutkan bahwa Pemerintah dan pemerintah daerah sesuai dengan kewenangannya dapat memberi beasiswa kepada peserta didik yang berprestasi. Menurut Agus Lahinta, beasiswa adalah pemberian berupa bantuan keuangan yang diberikan kepada perorangan yang bertujuan untuk digunakan demi keberlangsungan pendidikan yang ditempuh (Utomo, 2011).

Jadi dapat disimpulkan bahwa beasiswa merupakan bantuan berupa pendanaan bagi Pelajar / Mahasiswa yang kurang mampu dari segi ekonomi, maupun bagi yang berprestasi.

2. Sistem Informasi

Sistem dapat didefinisikan sebagai kumpulan dari elemen-elemen berupa data, jaringan kerja dari prosedur-prosedur yang saling berhubungan, sumber daya manusia, teknologi baik hardware maupun software yang saling berinteraksi sebagai satu kesatuan untuk mencapai tujuan/sasaran tertentu yang sama (Maniah

& Hamidin, 2017). Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan kegiatan atau untuk melakukan sasaran tertentu (Jeperson, 2014).

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi penerimanya (Jeperson, 2014). Secara umum informasi dapat didefinisikan sebagai hasil dari pengolahan data dalam suatu bentuk yang lebih berguna bagi penerimanya yang menggambarkan suatu kejadian-kejadian yang nyata yang digunakan untuk pengambilan keputusan (Irmawati & Indrihapsari, 2014). Informasi yang dibtuhkan tidak dilihat dari jumlah informasi yang dihasilkan, tetapi kualitas dari informasi (*quality of information*) tersebut karena tidak semua informasi berkualitas. Oleh sebab itu, sudah seharusnya dilakukan penyaringan terhadap informasi yang beredar atau yang dapat ditangkap (Ramadhania, 2015)

Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengelolaan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang dibutuhkan (Jeperson, 2014). Sistem informasi juga dapat diartikan sebagai suatu alat untuk menyajikan informasi dengan cara sedemikian rupa, sehingga bermanfaat bagi penerimanaya (Muslihudin & Oktafianto, 2016).

Sistem informasi terdiri dari komponen-komponen yang disebut blok bangunan (*building block*), yang terdiri dari komponen *input*, komponen *model*, komponen *output*, komponen teknologi, komponen *hardware*, komponen software,

komponen basis data, dan komponen kontrol (Jeperson, 2014). Menurut Jeperson, semua komponen tersebut saling berinteraksi satu dengan yang lain membentuk suatu kesatuan untuk mencapai sasaran. Berikut merupakan penjelasan dari masing-masing komponen *building block* (Jeperson, 2014) :

a. Blok masukkan (*input block)*

Input mewakili data yang masuk ke dalam sistem informasi. Input termasuk dalam metode-metode dan media yang digunakan untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen dasar.

b. Blok model (*model block*)

Blok ini terdiri dari kombinasi prosedur, logika dan metode matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang sudah diinginkan.

c. Blok keluaran (*output block*)

Produk dari sistem informasi adalah keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai system.

d. Blok teknologi (*technology block*)

Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian diri secara keseluruhan. Teknologi terdiri dari unsur utama :

1) Teknisi (*human ware* atau *brain ware*)
2) Perangkat lunak (*software*)
3) Perangkat keras (*hardware*)

e. Blok basis data (*data base block*)

Merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan diperangkat keras computer dan digunakan perangkat lunak untuk memanipulasinya.

f. Blok kendali (*control block*)

Banyak faktor yang dapat merusak sistem informasi, misalnya bencana alam, api, temperatur tinggi, air, debu, kecurangan-kecurangan, kejanggalan sistem itu sendiri, kesalahan-kesalahan ketidakefisienan, sabotase dan sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah atau bila terlanjur terjadi kesalahan dapat langsung diatasi.

3. Metode Pengembangan Sistem

a. Pengembangan Perangkat Lunak

Pengembangan perangkat lunak adalah disiplin teknik yang berkaitan dengan semua aspek produksi perangkat lunak mulai dari konsepsi awal hingga operasi dan pemeliharaan (Sommerville, 2016). Pengembangan perangkat lunak adalah sebuah disiplin dimana dalam menghasilkan perangkat lunak bebas dari kesalahan dan dalam pengiriman anggaran dapat tepat waktu serta memuaskan keinginan pemakai (Scach, 2011).

b. *Procedural Programming* & *Object Oriented Programming*

Procedural programming atau lebih dikenal dengan metode pemrograman tradisional merupakan metodologi yang menguraikan program ke dalam berbagai prosedur untuk menyelesaikan sebuah masalah. Sedangkan *object oriented*

programming merupakan metode yang medukung peningkatan dalam hal *class*, hubungan antar *class* dan *inheritence* yang dinamis (Ganesh, 2007). Berikut merupakan tabel perbedaan secara spesifik antara *procedural programming* dan *object oriented programming.*

Tabel 1. Perbedaan *Procedural Programming* dan *OOP*

No	***Procedural***	***OOP***
1	Fokus utama pada fungsi dan prosedur yang beroperasi pada data	Menekankan pada data yang sedang beroperasi
2	Program besar terbagi dalam program unit kecil yang disebut fungsi	Program dibagi ke dalam apa yang disebut objek
3	Data dari fungsi diperlakukan sebagai entitas sama	Data dari fungsi diperlakukan sebagai entitas terpisah
4	Data bebas bergerak disekitar sistem dari satu fungsi lain	Data tersembunyi dan tidak dapat diakses oleh objek eksternal
5	Data bersifat pasif	Objek-objeknya bersifat aktif
6	Program di desain dengan pendekatan *"Top-Down"* yaitu tugas-tugas kompleks dipecah menjadi bagian yang lebih kecil sampai sub-tugas tersebut mudah diimplementasikan	Program di desain dengan pendekatan *"Bottom-Up"* yaitu memuat prosedur-prosedur untuk menyelesaikan tugas-tugas sederhana, kemudian menggabungkan prosedur-prosedur tersebut dalam prosedur yang lebih kompleks, sampai masalah terpecahkan

Kesimpulannya adalah, OOP menyediakan tingkat abstraksi yang lebih tinggi dibandingkan *procedural programming,* dimana *procedural programming* mendukung pembuatan perangkat lunak untuk memecahkan masalah tertentu, sedangkan OOP mendukung pembuatan perangkat lunak yang lebih mudah untuk dikembangkan, digunakan kembali dan dipelihara.

c. Rational Unified Proccess

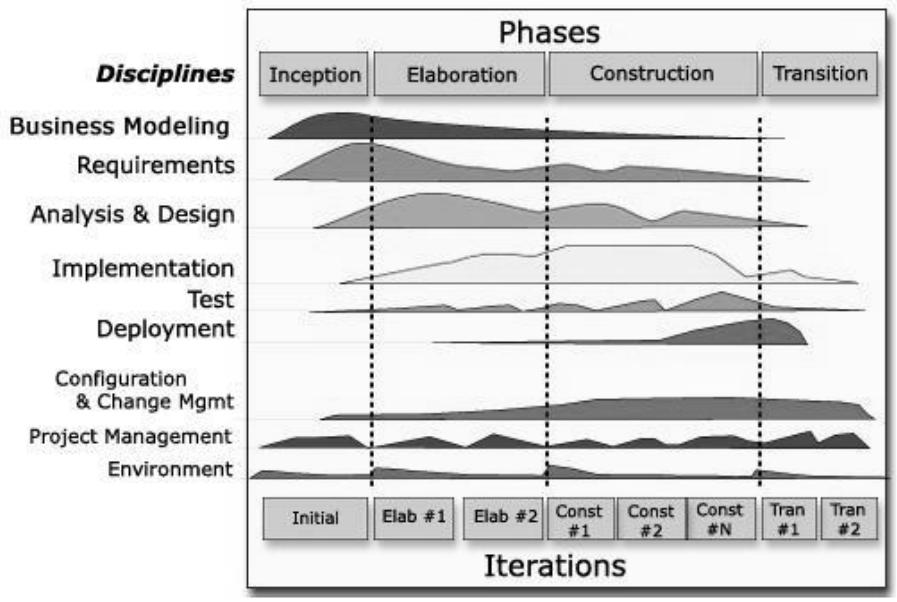
Menurut Rosa A. S. & Salahudin, *Unified Process* adalah sebuah proses dimana pengembangan perangkat lunak dilakukan secara iteratif dan inkremental. Iterasi

bisa dilakukan di dalam setiap tahap untuk menghasilkan perbaikan fungsi inkremetal dimana setiap iterasi akan memperbaiki iterasi berikutnya (Rosa & Shalahudin, 2011). Salah satu *Unified Process* yang terkenal yaitu *Rational Unified Process. Rational unified proccess* merupakan salah satu proses rekayasa perangkat lunak yang bersifat iteratif, berpusat pada arsitektur dan menggunakan *use case* untuk mendorong terjadinya pengerjaan pembangunan perangkat lunak, mulai dari pengumpulan awal dan penentuan kebutuhan sampai proses pembuatan program (Kroll & Kruchten, 2003). Menurut Anwar, Rational Unified Process adalah sebuah kerangka proses pengembangan perangkat lunak secara berulang yang dibuat oleh the Rational Software Corporation, sebuah divisi di IBM (Anwar, 2014).

Dikutip dari situs resmi IBM Developer Works, RUP menggunakan konsep object oriented, dengan aktifitas yang berfokus pada pengembangan model dengan menggunakan Unified Model Language (UML). Melalui gambar dibawah dapat dilihat bahwa RUP memiliki, yaitu:

1) Dimensi pertama digambarkan secara horizontal. Dimensi ini mewakili aspek-aspek dinamis dari pengembangan perangkat lunak. Aspek ini dijabarkan dalam tahapan pengembangan atau fase. Setiap fase akan memiliki suatu major Conmilestone yang menandakan akhir dari awal dari phase selanjutnya. Setiap phase dapat berdiri dari satu beberapa iterasi. Dimensi ini terdiri atas Inception, Elaboration, Construction, dan Transition.
2) Dimensi kedua digambarkan secara vertikal. Dimensi ini mewakili aspek-aspek statis dari proses pengembangan perangkat lunak yang dikelompokkan ke dalam beberapa disiplin. Proses pengembangan perangkat lunak yang

dijelaskan kedalam beberapa disiplin terdiri dari empat elemen penting, yakni who is doing, what, how dan when. Dimensi ini terdiri atas Business Modeling, Requirement, Analysis and Design, Implementation, Test, Deployment, Configuration dan Change Manegement, Project Management, Environtment.



Gambar 3. Dimensi dalam RUP

4. *Unified Modelling Language*

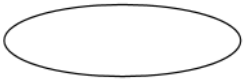
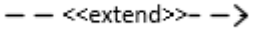
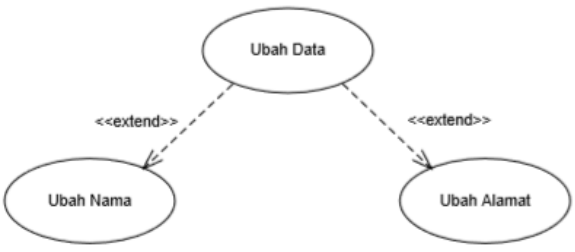
Unified modelling language adalah bahasa standar untuk menulis cetak biru perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan artefak sistem perangkat lunak yang intensif (Pressman & Maxim, 2014). Menurut Sommerville, UML adalah bahasa grafis yang digunakan dalam pengembangan berorientasi objek yang mencakup beberapa jenis model sistem yang memberikan pandangan berbeda dari suatu sistem (Sommerville, 2016). Pada tahap desain penelitian ini digunakan *use case diagram* dan *sequence diagram*.

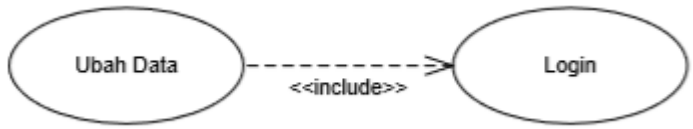
a. *Use Case Diagram*

Use case diagram digunakan untuk membantu menentukan fungsi dan fitur perangkat lunak dari perspektif pengguna. *Use case diagram* menggambarkan bagaimana seorang pengguna berinteraksi dengan sistem dengan menentukan langkah-langkah yang diperlukan untuk mencapai tujuan tertentu (Pressman & Maxim, 2014).

Simbol-simbol yang terdapat pada *use case* ditunjukkan pada tabel berikut ini

Tabel 2. Simbol dalam *use case diagram*

Simbol	Keterangan
	*Use case* merupakan fungsi dari sistem yang saling berhubungan / bertukar pesan. *Use case* dinyatakan dengan keta kerja.
	Aktor merupakan orang, proses atau sistem lain yang berhubungan dengan sistem yang dibuat. Aktor dinyatakan dengan kata benda.
	Association merupakan relasi tambahan antar *use case* atau *use case* dengan aktor
	*Extend* merupakan relasi tambahan antar *usecase. Usecase* yang ditambahkan dapat berdiri sendri tanpa *use case* tambahan. Arah panah menunjuk pada *usecase* tambahan.  Dari gambar diatas dapat disimpulkan bahwa ubah data merupakan *usecase* utama dan ubah nama dan ubah alamat merupakan *use case* tambahan. *Use case* ubah data meng-*extend* ubah nama dan ubah alamat sehingga *usecase* ubah data dapat berjalan walau tanpa *use case* ubah nama dan ubah alamat.

<<include>>	*Include* adalah relasi tambahan antar *use case. Use case* yang ditambahkan memerlukan *use case* utama untuk menjalankan fungsinya. Arah panah mengarah ada *usecase* yang harus berjalan terlebih dahulu.  Berdasarkan gambar diatas dapat disimpulkan bahwa *usecase* login meng-*include usecase* ubah data, sehingga *usecase* ubah data akan berjalan jika login sudah dijalankan terlebih dahulu.

b. *Sequence Diagram*

Sequence diagram menunjukkan pemanggilan metode menggunakan panah horizontal dari pemanggil ke objek yang dipanggil, diberi label dengan nama metode dan secara opsional termasuk parameternya, jenisnya, dan jenis kembalinya (Pressman & Maxim, 2014). Menurut Sommerville, *sequence diagram* digunakan untuk memodelkan interaksi antara aktor dan objek dalam suatu sistem dan interaksi antara objek itu sendiri, fokusnya adalah pada dasar-dasar tipe diagram ini (Sommerville, 2016).

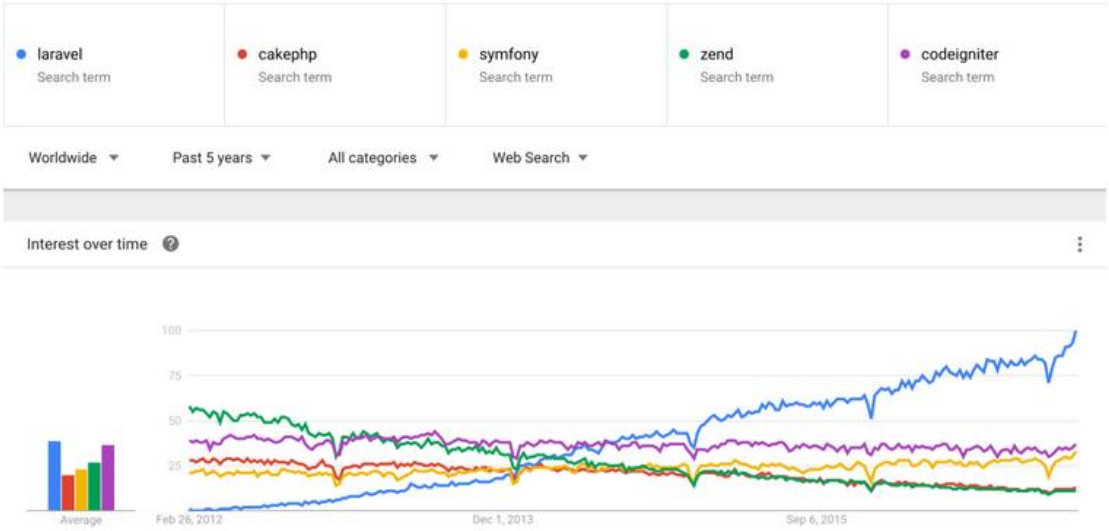
5. Laravel Framework

Laravel adalah sebuah *framework* PHP yang dirilis dibawah lisensi MIT, dibangun dengan konsep MVC (model view controller). Laravel adalah pengembangan website berbasis MVC yang ditulis dalam PHP yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, dan untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan sintaks yang ekspresif, jelas dan menghemat waktu.

MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. MVC memisahkan aplikasi berdasarkan komponen-komponen aplikasi, seperti : manipulasi data, controller, dan user interface.

- Model mewakili struktur data. Model berisi fungsi-fungsi yang membantu programmer dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
- View adalah bagian yang mengatur tampilan ke pengguna. Bisa dikatakan view berupa halaman web
- Controller merupakan bagian untuk menjembatani bagian model dan view dari sebuah framework

Menurut data survey yang dikutip dari sitepoint.com, menunjukkan bahwa pada tahun 2017, Laravel masih menjadi yang paling populer dikalangan pengembang *web application*



Gambar 4. Hasil survey penggunaan framework PHP

(sumber : https://www.sitepoint.com/the-state-of-php-mvc-frameworks-in-2017/)

Keuntungan dari menggunakan *framework* Laravel yaitu :

- Sebuah aplikasi web menjadi lebih terukur
- Menghemat waktu dalam pengerjaan aplikasi berbasis web, karena laravel dapat memanfaatkan kembali komponen dari *framework* yang lain
- Dalam Laravel terdapat *namespace* dan *interfaces,* sehingga lebih mudah dalam manjemen *source code*

6. Pengujian Perangkat Lunak

Rosa A.S dan Shalahuddin mengungkapkan bahwa pengujian perangkat lunak merupakan satu set aktifitas yang direncanakan dan sistematis untuk menguji atau mengevaluasi kebenaran yang dinginkan (Rosa & Shalahudin, 2011). Pengujian perangkat lunak dilakukan untuk mengurangi kesalahan secara teknis dan juga non teknis. Menurut Roger S. Pressman, Pengujian adalah serangkaian kegiatan yang dapat direncanakan sebelumnya dan dilakukan secara sistematis. Pengujian merupakan elemen dari pengembangan perangkat lunak yang disebut dengan *verification and validation testing (V&V).* Verifikasi mengacu pada serangkaian kegiatan yang memastikan perangkat lunak dapat menjalankan fungsi yang telah ditentukan, sedangkan validasi mengacu pada satu set aktifitas yang memastikan bahwa perangkat lunak yang dikembangkan sudah sesuai dengan kebutuhan pengguna (Pressman & Maxim, 2014).

7. Tahap Pengujian Perangkat Lunak

Tahapan pengujian yang dilakukan untuk menguji perangkat lunak adalah sebagai berikut :

a. *Unit Testing*

Unit testing adalah proses pengujian komponen program, seperti metode atau kelas objek. Fungsi atau metode individu adalah jenis komponen yang paling sederhana (Sommerville, 2016). Tahap pengujian ini berfokus pada upaya verifikasi unit terkecil (komponen / modul perangkat lunak). Untuk menguji unit-unit perangkat lunak, biasanya dibuat prosedur atau fungsi, sekumpulan prosedur atau fungsi yang ada dalam satu file (pemrograman terstruktur), berupa kelas atau sekumpulan kelas dalam satu *package* (pemrograman berorientasi objek). *Unit testing* ini biasanya dilakukan saat *coding* (Pressman & Maxim, 2014).

b. *Integration Testing*

Pengujian integrasi adalah teknik sistematis untuk membangun arsitektur perangkat lunak sementara pada saat yang sama melakukan tes untuk mengungkap kesalahan yang terkait dengan interfacing. Tujuannya adalah untuk mengambil komponen yang telah teruji unit dan membangun struktur program yang telah ditentukan oleh desain. Pengujian integrasi lebih difokuskan pada pengujian penggabungan unit yang terdapat pada perangkat lunak. Menurut Pressman, *integration testing* sebaiknya dilakukan secara bertahap untuk menghindari kesulitan penelusuran jika terjadi *error* (Pressman & Maxim, 2014).

c. *System Testing*

System testing memeriksa bahwa komponen kompatibel, berinteraksi dengan benar, dan mentransfer data yang tepat pada waktu yang tepat di seluruh antarmuka mereka (Sommerville, 2016). Pengujian sistem berfungsi untuk memverifikasi

bahwa elemen-elemen sistem telah terintegrasi dengan benar dan melakukan fungsi-fungsi yang dialokasikan (Pressman & Maxim, 2014). *System testing* terdiri dari :

1) *Recovery Testing*

Recovery testing adalah pengujian sistem yang memaksa perangkat lunak gagal dalam berbagai cara dan memverifikasi bahwa pemulihan dilakukan dengan benar. Jika pemulihan bersifat otomatis (dilakukan oleh sistem itu sendiri), *reinitialization*, mekanisme pemeriksaan, pemulihan data, dan *restart* maka sistem sudah diklasifikasi layak. Jika pemulihan membutuhkan intervensi manusia, *mean-time-to-repair* (MTTR) dievaluasi untuk menentukan apakah itu dalam batas yang dapat diterima.

2) *Security Testing*

Security testing dilakukan untuk memverifikasi bahwa mekanisme perlindungan yang dibangun ke dalam suatu sistem akan, pada kenyataannya, melindunginya dari penetrasi yang tidak tepat

3) *Stress Testing*

Stress Testing mengeksekusi suatu sistem dengan cara yang menuntut sumber daya dalam jumlah, frekuensi, atau volume yang abnormal. Sehingga dapat diketahui *stress-level* dari sistem yang telah dikembangkan.

4) *Performance Testing*

Performance testing sering digabungkan dengan pengujian tegangan dan biasanya membutuhkan instrumentasi hardware dan software. Dengan menginstruksikan suatu sistem, tester dapat mengungkap situasi yang menyebabkan degradasi dan kemungkinan kegagalan sistem.

5) *Deployment Testing*

Deployment testing dilakukan dengan menjalankan perangkat lunak di setiap lingkungan tempat ia beroperasi. Selain itu, deployment testing memeriksa semua prosedur pemasangan dan *software installer* khusus yang akan digunakan oleh pelanggan dan semua dokumentasi yang akan digunakan untuk memperkenalkan perangkat lunak kepada pengguna.

8. Jenis Pengujian Perangkat Lunak

a. *White-Box Testing*

Pengujian *white-box* adalah metode desain pengujian yang menggunakan struktur kontrol yang digambarkan sebagai bagian dari desain tingkat komponen untuk menurunkan kasus uji. Dengan menggunakan metode pengujian white-box, dapat diturunkan menjadi kasus uji yang menjamin bahwa semua jalur independen dalam modul telah dilakukan setidaknya sekali, menjalankan semua keputusan logis di sisi mereka yang benar dan salah, menjalankan semua *loop* pada batas-batas mereka dan dalam batas-batas operasional mereka, dan latihan struktur data internal untuk memastikan validitasnya (Pressman & Maxim, 2014).

b. *Black-Box Testing*

Black-box testing berfokus pada persyaratan fungsional perangkat lunak. Pada teknik pengujian *black-box* dimungkinkan untuk menentukan set kondisi input yang akan sepenuhnya menjalankan semua persyaratan fungsional untuk suatu program. Pengujian *black-box* bukan merupakan alternatif teknik *white-box*. Sebaliknya, itu adalah pendekatan pelengkap yang kemungkinan akan mengungkap kelas kesalahan yang berbeda dari metode *white-box* (Pressman & Maxim, 2014).

c. *Alpha Testing*

Dalam pengujian alfa, pengguna dan pengembang bekerja sama untuk menguji sistem saat sedang dikembangkan. Pengguna dapat mengidentifikasi masalah yang tidak mudah terlihat oleh tim penguji pengembangan. Oleh karena itu, pengguna dapat memberikan informasi tentang praktik yang membantu dengan desain tes yang lebih realistis (Sommerville, 2016).

d. *Beta Testing*

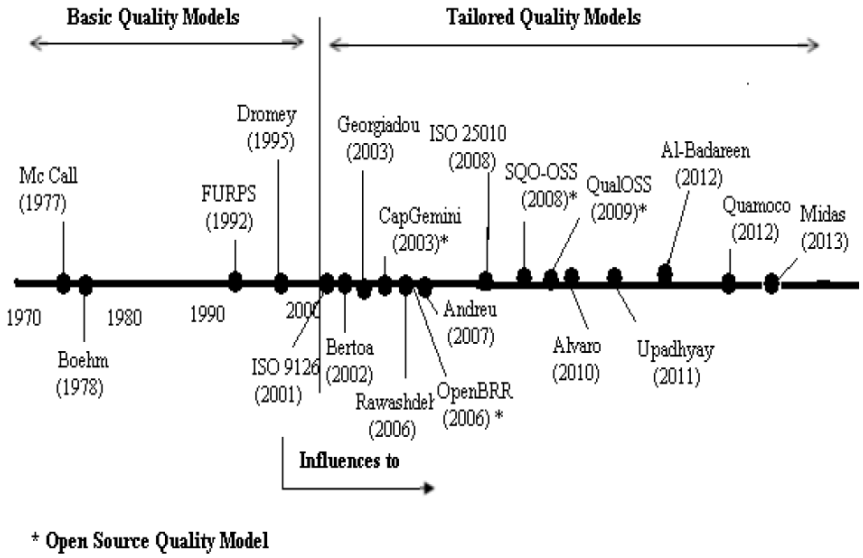
Beta testing dilakukan ketika rilis sistem perangkat lunak yang lebih awal, terkadang sebelum selesai dan tersedia bagi kelompok pelanggan dan pengguna yang lebih besar untuk evaluasi. Penguji beta mungkin merupakan kelompok pelanggan terpilih yang merupakan pengguna awal sistem ini (Sommerville, 2016).

9. Analisis Kualitas Perangkat Lunak

Kualitas perangkat lunak dapat didefinisikan sebagai Proses perangkat lunak yang efektif diterapkan dengan cara tertentu yang menciptakan produk yang bermanfaat yang memberikan nilai terukur bagi mereka yang memproduksinya (Pressman & Maxim, 2014). Kualitas perangkat lunak adalah produk yang

dikembangkan harus sesuai dengan kebutuhan dan spesifikasi penggunanya (Sommerville, 2016). Kualitas merupakan persyaratan penting untuk kelangsungan hidup sebuah sistem. Untuk memastikan kualitas sistem, terdapat banyak model evaluasi kualitas dikembangkan oleh peneliti, hingga membantu menentukan kualitas calon sistem (Isaias & Issa, 2015).

Terdapat berbagai macam standar pengujian kualitas software, diantaranya Mc Call, Boehm, FURPS, Dromey dan ISO 9126 yang kemudian pada tahun 2017 digantikan oleh ISO 25010 atau yang lebih dikenal dengan SQuaRE (*Software engineering - Software product Quality Requirements and Evaluation*). (Miguel, Mauricio, & Rodriguez, 2014).



Gambar 5. *Open Source Quality Models*

(sumber : A Review of Software Quality Models for the Evaluation of Software Products)

Beberapa standar pengujian yang dipaparkan diatas, ISO 25010 merupakan standar pengujian internasional dasar, sehingga digunakan ISO 25010 untuk

menguji sistem yang dikembangkan. Terdapat 8 karakteristik dalam ISO 25010, yaitu *functional suitability*, *reliability*, *performance efficiency*, *usability*, *security*, *compatibility*, *maintainability*, dan *portability*.

Gambar 6. Karakteristik ISO 25010

(sumber : http://iso25000.com/index.php/en/iso-25000-standards/iso-25010)

Menurut Olsina dan kolega (Pressman & Maxim, 2014) standar kualitas aplikasi berbasis web dinilai dari lima aspek yaitu *usability*, *functionality*, *reliability*, *efficiency* dan *maintainability* dengan perincian sebagai berkut :

Tabel 3. Standar aplikasi web menurut Olsina dkk

No	Aspek	Keterangan
1	*Functionality*	Kemampuan pencarian dan penerimaan
		Fitur-fitur navigasi dan perambahan
		Fitur-fitur aplikasi yang berhubungan dengan ranah
2	*Usability*	Kemudahan pemahaman situs global
		Umpan balik dari pengguna dan fitur bantuan
		Antarmuka pengguna dan fitur estetika
		Fitur khusus
3	*Reliability*	Pembenahan pemrosesan tautan
		Pemulihan dari kesalahan
		Validasi dan pemulihan asupan pengguna
4	*Efficiency*	Kinerja waktu tanggap aplikasi web
		Kecepatan memuat halaman
		Kecepatan penggambaran grafik
5	*Maintainability*	Kemudahan melakukan koreksi
		Kemampuan aplikasi web untuk beradaptasi
		Kemampuan aplikasi web untuk dikembangkan

Berdasar standar kualitas web menurut Olsina dkk, untuk menguji kualitas sistem yang dikembangkan hanya akan mengambil 5 karakteristik ISO 25010 yaitu *functional suitability, usability, reliability, performance efficiency* dan *maintainability* dengan perbandingan sebagai berikut :

Tabel 4. Perbandingan standar kualitas aplikasi web Olsina dan ISO 25010

Standar *Web App* Olsina	**ISO 25010**
Functionality	*Functional suitability*
Usability	*Usability*
Reliability	Reliability
Efficiency	*Performance efficiency*
Maintainability	*Maintainability*

Berikut ini penjabaran dari 5 karakteristik yang digunakan untuk menguji kualitas dari sistem informasi beasiswa yang dikembangkan :

a. *Functional suitability*

Karakteristik sejauh mana suatu produk atau sistem yang memenuhi kebutuhan ketika digunakan pada kondisi tertentu. Karakteristik ini dibagi menjadi beberapa subkarakteristik yaitu:

1) *Functional completeness*, sejauh mana fungsi yang disediakan mencakup semua tugas dan tujuan pengguna secara spesifik.
2) *Functional correctness*, sejuah mana produk atau sistem menyediakan hasil yang benar sesuai kebutuhan.
3) *Functional appropriateness*, sejauh mana fungsi yang disediakan mampu memfasilitasi penyelesaian tugas dan tujuan tertentu.

b. *Usability*

Karakteristik sejuah mana sebuah produk atau sistem dapat digunakan oleh pengguna tertentu untuk mencapai tujuan dengan efektif, efisien, dan kepuasan tertentu dalam konteks pengguna. Karakteristik ini terbagi menjadi beberapa subkarakteristik yakni sebagai berikut:

1) *Appropriariateness recognizability*, karakter sejauh mana pengguna dapat mengetahui apakah sistem atau produk sesuai kebutuhan mereka.
2) *Learnability*, karakteristik sejauh mana produk atau sistem dapat digunakan oleh pengguna untuk mencapai tujuan tertentu yang belajar menggunakan sistem atau produk dengan efisien, efekif, bebas dari resiko, dan mendapatkan kepuasan dalam konteks tertentu.
3) *Operability*, karakteristik sejauh mana produk atau sistem mudah dioperasikan.
4) *User error protection*, karakteristik sejauh mana produk atau sistem melindungi pengguna terhadap kesalahan penggunaan.
5) *User interface aesthetics*, sejauh mana antarmuka pengguna dari produk atau sistem memungkinkan interaksi yang menyenangkan dan memuaskan pengguna.
6) *Accessbility*, sejauh mana produk atau sistem dapat digunakan oleh semua kalangan untuk mencapai tujuan tertentu sesuai konteks penggunaan.

c. *Reliability*

Karakteristik sejauh mana sistem, produk, atau komponen melakukan fungsi tertentu di bawah kondisi tertentu dalam jangka waktu yang ditetapkan. Karakteristik ini terbagi menjadi beberapa subkarakteristik, yaitu:

1) *Maturity*, sejauh mana produk atau sistem mampu memenuhi kebutuhan secara handal di bawah keadaan normal.
2) *Availability*, sejauh mana produk atau sistem siap beroperasi dan dapat diakses saat perlu digunakan.
3) *Fault tolerence*, sejauh mana produk atau sistem tetap berjalan sebagaimana yang dimaksud meskipun terjadi kesalahan pada perangkat keras atau perangkat lunak.
4) *Recoverability*, sejauh mana produk atau sistem mampu dapat memulihkan data yang terkena dampak secara langsung dan menata ulang kondisi sistem seperti yang dinginkan ketika terjadi gangguan.

d. *Performance efficiency*

Tingkat kinerja relatif terhadap sumber daya yang digunakan dalam kondisi yang ditetapkan. Karakteristik ini terbagi menjadi beberapa subkarakteristik yaitu:

1) *Time behaviour*, sejuah mana respon dan pengolahan waktu produk atau sistem dapat memenuhi persyaratan ketika menjalankan fungsi.
2) *Resource utilization*, sejauh mana jumlah dan jenis sumber daya yang digunakan oleh produk atau sistem dapat memenuhi persyaratan ketika menjalankan fungsi.
3) *Capacity*, sejauh mana batas maksimum parameter produk atau sistem dapat memenuhi persyaratan.

e. *Maintainabiity*

Tingkat efektivitas dan efisiensi pada suatu produk atau sistem untuk dapat dimodifikasi oleh pengembang. Karakteristik ini terbagi menjadi beberapa subkarakteristik yaitu:

1) *Modularity*, sejauh mana sistem terdiri dari komponen terpisah sehingga perubahan atau modifikasi pada salah satu komponen tersebut memiliki dampak yang kecil terhadap komponen yang lain.
2) *Reusability*, sejauh mana aset dapat digunakan oleh satu sistem atau digunakan untuk membangun aset lain.
3) *Analyzability*, tingkat efektivitas dan efisiensi untuk mengkaji dampak perubahan pada satu atau lebih bagian-bagian produk atau sistem, untuk mendiagnosis kekurangan atau penyebab kegagalan produk, untuk mengidentifikasi bagian yang akan diubah.
4) *Modifiability*, sejauh mana produk atau sistem dapat dimodifikasi secara efektif dan efisien tanpa menurunkan kualitas produk yang ada.
5) *Testability*, tingkat efektivitas dan efisiensi untuk membentuk kriteria uji dari produk, sistem atau komponen dan uji dapat dilakukan untuk menenukan apakah kriteria tersebut terpenuhi.

B. Penelitian yang Relevan

Hasil penelitian yang relevan antara lain :

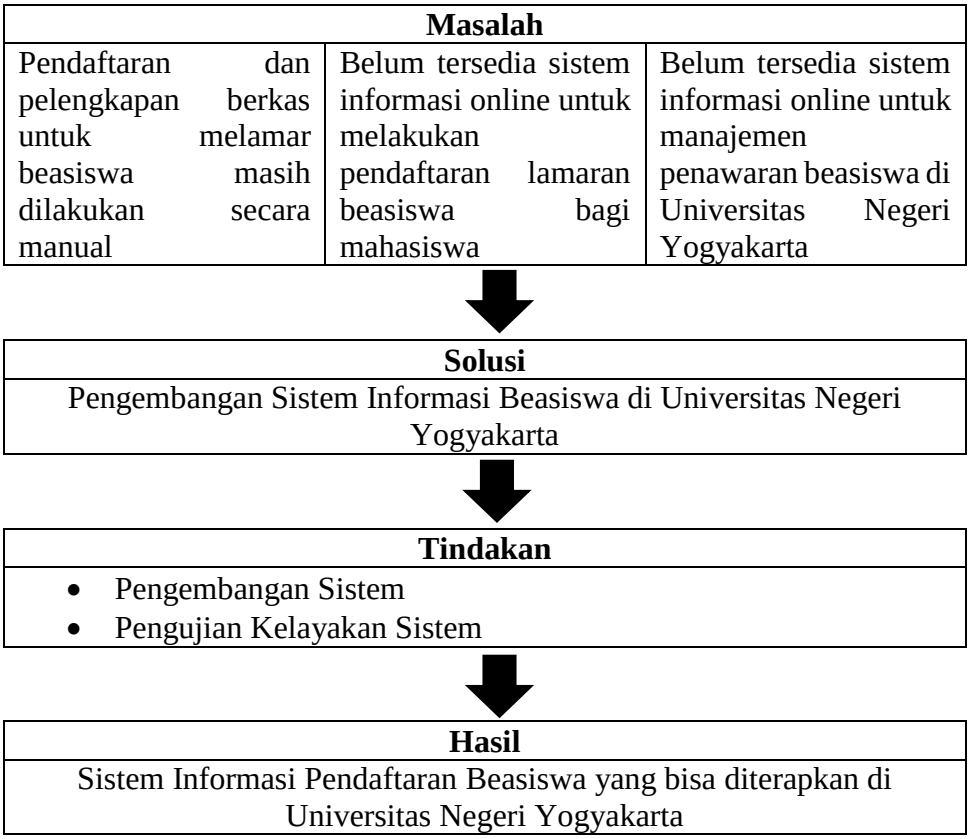
1. Pengembangan Sistem Informasi Beasiswa Fakultas Teknik Universitas Negeri Yogyakarta yang dilakukan oleh Ringan Oktiah (2011) dengan hasil penelitian menunjukkan tingkat kelayakan oleh ahli database dan pemrograman sebesar 74,54% yang dikategorikan layak, ahli sistem informasi sebesar 80,00% yang

dikategorikan layak, petugas sebesar 74,75% yang dikategorikan layak, mahasiswa sebesar 74,35% yang dikategorikan layak. Relevansi penelitian ini yaitu mengenai konsep pengembangan sistem informasi beasiswa di Fakultas Teknik Universitas Negeri Yogyakarta berbasis web. Kekurangan penilitian ini yaitu belum menggunakan *framework* dalam pengembangannya sehingga sulit untuk proses *maintenance.*

2. Analisis dan Pengembangan Sistem Informasi Monitoring Beasiswa Bidikmisi Fakultas Teknik Universitas Negeri Yogyakarta yang dilakukan oleh Bayu Setiawan (2015) relevansi dengan penelitian berikut yaitu pada pengembangan sistem beasiswa, namun pada penelitian ini hanya menakankan pada pemantauan mahasiswa penyandang status bidik misi, tidak tersedia untuk beasiswa umum. Pengujian kualitas sistem berdasarkan ISO 9126 pada aspek *functionality, usability* dan *portability* dengan hasil uji aspek *functionality* sebesar 1 (baik) dan *security* menggunakan *Acunetix Web Vulnerability Scanner* pada level 2 atau *medium*. Sedangkan untuk aspek *usability* sebesar 75,89% (tinggi) dengan *alpha cronbach* sebesar 0,935 (*excellent*) dan aspek *portability* yang terpenuhi dengan menjalankan di 5 *web browser* tanpa error.

C. Kerangka Pikir

Sistem informasi beasiswa ini merupakan tempat untuk memudahkan dalam penyaluran informasi paket beasiswa yang tersedia dan tempat untuk mempermudah Mahasiswa dalam melamar beasiswa di Universitas Negeri Yogyakarta. Kerangka berfikir dari Sistem Informasi Beasiswa Universitas Negeri Yogyakarta digambarkan pada gambar dibawah :



D. Pertanyaan Penelitian

Berdasarkan uraian diatas, maka dapat dirumuskan pertanyaan penelitian sebagai berikut :

1. Apakah sistem informasi beasiswa memenuhi karakteristik *functional suitability?*
2. Apakah sistem informasi beasiswa memenuhi karakteristik *usability?*
3. Apakah sistem informasi beasiswa memenuhi karakteristik *reliability?*
4. Apakah sistem informasi beasiswa memenuhi karakteristik *performance efficiency?*
5. Apakah sistem informasi beasiswa memenuhi karakteristik *maintainability?*