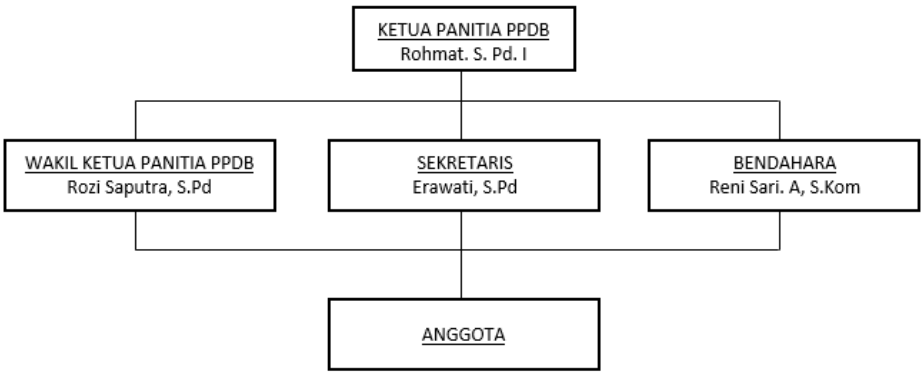
BAB 2
LANDASAN TEORI

2.1 Yayasan Dear Teknologi Hamid Pekanbaru

Yayasan Dear Teknologi Hamid (YDTH) merupakan instansi yang bergerak di bidang pendidikan yang terdiri dari SD, SMP, SMA yang beralamat di Jl. Budi Daya RT 01 RW 07 Kelurahan Tuah Karya Kecamatan Tampan Provinsi Riau yang didrikan pada tanggal 26 Februari 2009. Pada tingkat SD, sekolah ini memiliki 17 tenaga kerja guru dan 12 kelas yang rata-rata setiap kelas di isi oleh 25 sampai dengan 30 siswa. Untuk tingkat SMP, sekolah ini memiliki 16 tenaga kerja guru dengan jumlah kelas sebanyak 9 kelas yang rata-rata setiap kelas di isi oleh 30 sampai dengan 35 siswa. Sedangkan untuk tingkat SMA, sekolah ini memiliki dua jurusan yaitu Ilmu Pengetahuan Alam (IPA) dan Ilmu Pengetahuan Sosial (IPS) dengan 11 tenaga kerja guru dan mempunyai 12 kelas yang rata-rata di isi dengan 30 sampai dengan 35 siswa.

2.1.1 Struktur Panitia PPDB Yayasan Dear Teknologi Hamid Pekanbaru

Secara garis besar struktur panitia PPDB Yayasan Dear Teknologi Hamid Pekanbaru dapat dilihat pada Gambar 2.1.



Gambar 2.1. Struktur panitia PPDB Yayasan Dear Teknologi Hamid Pekanbaru

2.1.2 Tugas dan Tanggung Jawab

Adapun uraian tugas dari Panitia PPDB Yayasan Dear Teknologi Hamid Pekanbaru sebagai berikut:

1. Ketua Panitia PPDB

   Adapun tugas pokok ketua paniata PPDB adalah:

   (a) Penanggung jawab umum pelaksanaan kegiatan.

   (b) Membentuk panitia kegiatan.

(c) Membentuk SK kepanitiaan.

2. Wakil Ketua PPDB

   Adapun tugas pokok wakil ketua PPDB adalah bertanggung jawab tersedianya administrasi penyelenggaraan.

3. Sekretaris

   Adapun tugas pokok sekretaris adalah mengkoordinir kegiatan persiapan administrasi penyelenggaraan yang meliputi administrasi pelaksanaan kesektretariatan dan pelaporan.

4. Bendahara

   Adapun tugas bendahara adalah:

   (a) Mengelola dan bertanggung jawab atas penggunaan dana yang ada.
   (b) Membuat laporan keuangan PPDB.

5. Anggota

   Adapun tugas pokok anggota adalah:

   (a) Menyiapkan berkas-berkas PPDB.
   (b) Mengelola dan mengarsipkan berkas-berkas PPDB.
   (c) Membuat laporan PPDB.
   (d) Menyiapkan konsumsi panitia.

2.2 Rancang Bangun

Perancang atau merancang merupakan serangkaian prosedur untuk menterjemahkan hasil analisis dan sebuah sistem kedalam bahasa pemograman untuk mendeskripsikan dengan detail bagaimana komponen-komponen sistem di implementasikan. Sedangkan pengertian pembangunan atau bangun sistem adalah kegiatan menciptakan sistem baru maupun mengganti atau memperbaiki sistem yang telah ada baik seacara keseluruhan maupun bagian (Kumaladewi dan Elsy, 2014).

2.3 Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi, mendukung operasi, bersifat manajerial, dan kegiatan strategi dari suatu organisasi serta menyediakan pihak yang terkait dengan laporan-laporan yang diperlukan. Dalam sebuah organisasi tentunya diperlukan sebuah sistem yang mendukung jalannya kegiatan operasional yang ada di dalam organisasi tersebut, hal ini dapat teratasi dengan adanya sistem informasi di dalam organisasi tersebut. Terdapat beberapa komponen dalam sistem informasi, yang dapat diklasifikasikan sebagai berikut *hardware* dan *software* yang berfungsi sebagai mesin, *people* dan *procedures* yang merupakan manusia dan tata

cara menggunakan mesin, dan data merupakan jembatan penghubung antara manusia dan mesin agar terjadi suatu proses pengolahan data (Ladjamudin, 2005).

2.4 Sistem Informasi Berbasis *Web*

Sistem informasi dahulu dibuat secara konvensional (aplikasi desktop), namun seiring dengan perkembangan teknologi internet maka sistem informasi dibuat berbasis *web* karena sifatnya yang luas dan memungkinkan semua orang dapat mengakses informasi secara cepat dan mudah dari mana saja, sehingga pemasukan data dapat dilakukan dari mana saja dan dapat dikontrol dari satu tempat sebagai sentral. *world wide web* (*WWW*) atau yang biasa disingkat dengan *web*, ini merupakan salah satu bentuk layanan dapat diakses melalui internet (Azzaky, Chumaidiyah, dan Tripiawan, 2016).

2.5 PPDB *Online*

Penerimaan peserta didik baru (PPDB) *online* merupakan sistem yang dapat melakukan otomasi PPDB. Otomasi yang di maksud disini meliputi proses pendaftaran dan pengumuman hasil penerimaan peserta didik baru secara *online* dan *realtime*. Terdapat beberapa sisi positif dari penerapan/implementasi sistem PPDB *online* ini, diantaranya adalah calon peserta didik dan panitia PPDB lebih ringan dan ringkas kerjanya. Adapun hal positif lainnya dari penerapan PPDB *online* ini adalah para calon peserta didik hanya tinggal melihat *update* pengumuman hasil PPDB melaui internet dimana saja dan kapan saja.

2.6 *Database Server*

Database (Basis Data) adalah kumpulan dari beberapa data yang saling berhubungan antara satu dengan yang lainnya, dimana digunakan suatu *software* atau perangkat lunak yang ada pada komputer untuk memanipulasinya. Basis Data adalah kumpulan data yang saling berhubungan secara logikal serta deskripsi dari data tersebut, yang dirancang untuk memenuhi kebutuhan informasi suatu organisasi. Basis Data adalah sebuah penyimpanan data yang besar yang bisa digunakan oleh banyak pengguna dan departemen. Semua data terintegrasi dengan jumlah duplikasi yang minimum. Basis Data tidak lagi dipegang oleh satu departemen, tetapi dibagikan ke seluruh departemen pada perusahaan. Basis Data itu sendiri tidak hanya memegang data operasional organisasi tetapi juga penggambaran dari data tersebut (Connolly dan Begg, 2005).

2.7 Bahasa Pemrograman *Web*

Bahasa pemrograman *Web* merupakan suatu bahasa yang digunakan untuk membangun sebuah *website* yang nantinya akan diterjemahkan. Adapun bahasa

yang gunakan penulis untuk membuat *website* adalah sebagai berikut (Azzaky dkk., 2016):

1. PHP

   HyperText Preprocessor (PHP) merupakan bagian dari bahasa pemrograman *web* atau *script-script* yang membuat dokumen HTML secara *on the fly* yang dieksekusi di *server web*. PHP merupakan sebuah bahasa *scripting* komputer, pada awalnya didesain untuk menghasilkan halaman *web* yang dinamis, secara umum digunakan sebagai *script server-side*, tetapi dapat digunakan dari *interface command line* atau secara berdiri sendiri sebagai aplikasi grafis.

2. HTML

   Hyper Text Markup Language (HTML) merupakan file teks atau file ASCII yang berisi instruksi/*script* kepada *web browser* untuk menampilkan suatu tampilan grafis dari sebuah halaman *web*. Pada file HTML ini terdapat "tag" atau kode-kode yang dapat dimengerti nantinya oleh *web browser*.

3. CSS

   CSS adalah suatu bahasa *stylesheet* yang mengatur tampilan suatu dokumen. Pada umumnya CSS digunakan untuk memformat halaman *web* yang ditulis dengan HTML dan XHTML. Dengan CSS, tampilan *website* akan lebih cantik dan konsisten.

2.8 *Responsive Web Design*

Responsive web design digunakan dengan tujuan untuk memastikan informasi *website* yang akan disampaikan berjalan dengan baik tanpa kehilangan informasi dan terlepas diakses dari perangkat mobile apapun. Menurut Jeffrey Zeldman yang dikutip oleh Hidayat, Utomo, dan Djohan (2016), *responsive web design* adalah sebuah teknik yang digunakan *designer website* untuk memberikan pengalaman *visual* yang elegan tanpa mempedulikan ukuran *browser* yang digunakan dan batasan apapun tentang cara mengakses perangkat tersebut. Sebuah desain dianggap *responsif* jika menggunakan tiga poin yaitu grid yang fleksibel, gambar dan media yang fleksibel, dan permintaan media. Dengan menggunakan permintaan media, desainer *website* dapat menentukan rentang resolusi tertentu sebagai kondisi untuk menggunakan definisi CSS tertentu yang disebut *fixed breakpoints*. Dengan cara itu, perancang dapat mencocokkan definisi CSS mana yang akan diterapkan untuk resolusi tertentu yang akan menciptakan pengalaman *visual* yang lebih baik kepada pengguna situs.

Menurut Frank Farris yang dikutip oleh Hidayat dkk. (2016), desain *respon-*

sif pada perangkat mobile menciptakan sebuah *website* yang memerlukan sedikit interaksi pengguna (*scroll and click*) dari pada *website nonresponsif* di perangkat mobile untuk mencapai tujuan yang sama. Manfaat dari *responsive web design* dikatakan demikian karena sebuah *website* yang dapat beradaptasi tata letaknya untuk ukuran *browser* harus mampu beradaptasi dengan ukuran *fon*t, gambar, dan komponen lainnya sehingga pengguna dapat membaca seluruh isi tanpa menggulung layar secara horisontal (*horizontal scrolling*) untuk melihat bagian yang tersembunyi dari *website*. Jumlah klik yang disebabkan oleh kesalahan harus berkurang karena *website responsif* sebenarnya dirancang untuk membuat antar muka pengguna yang nyaman dan menangani ukuran yang terbatas dari *mobile browser*.

2.9 *Object Oriented Analysis and Design* (*OOAD*)

2.9.1 *Object Oriented Analysis* (*OOA*)

OOA merupakan tahapan perangkat lunak dengan menentukan spesifikasi sistem atau *System Requirement Specification* (*SRS*) dan mengidentifikasi kelas-kelas serta hubungan satu terhadap yang lainnya. Proses memahami spesifikasi sistem, kita perlu mengidentifikasi para pengguna atau yang sering disebut sebagai aktor-aktor. Siapa aktor-aktor yang akan menggunakan sistem dan bagaimana mereka menggunakan sistem (Nugroho, 2005).

Mencari objek-objek fisik pada sistem juga memungkinkan kita untuk mendapatkan informasi lebih lengkap terhadap objek-objek pada sistem yang bersangkutan. Objek-objek dapat bersifat mandiri, organisasi, satuan informasi, gambar, atau apapun yang menyusun suatu aplikasi dalam konteks representasi dunia nyata dalam sistem yang sedang dikembangkan. Adapun aktifitas utama dari OOA adalah (Nugroho, 2005).

2.9.2 *Object Oriented Design* (*OOD*)

OOD adalah pekerjaan yang dilakukan untuk merancang kelas-kelas yang teridentifikasi selama tahap analisis dan antarmuka (*interface*). Selama tahap ini kita mengidentifikasi dan menambah beberapa objek dan kelas yang mendukung implementasi dari spesifikasi kebutuhan (Nugroho, 2005).

2.10 *Unified Modelling Language* (*UML*)

UML adalah sebuah alat bantu yang sangat handal di dunia pengembangan sistem berorientasi objek. Hal ini di sebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atas visi mereka dalam bentuk baku, mudah di mengerti serta dilengkapi dengan mekanisme efektif untuk berbagi dan mengkomunikasikan rancangan mereka yang lain (Nugroho, 2005).

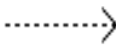
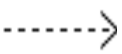
Metode pengembangan sistem yang digunakan yaitu desain menggunakan *unified modeling language* (UML). *unified modelling language* (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

Untuk merancang sebuah model, UML memiliki beberapa diagram antara lain: *use case diagram*, *class diagram*, *statechart diagram*, *activity diagram*, *sequence diagram*, *collaboration diagram*, *component diagram*, *deployment diagram*.

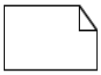
2.10.1 *Use Case Diagram*

Use case diagram merupakan suatu cara pemodelan untuk menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, *use case diagram* mendeskripsikan sebuah interaksi antara satu aktor atau lebih dengan sebuah sistem yang akan dibuat. Dalam artian bahwa sebuah *use case diagram* dapat digunakan untuk menggambarkan fungsi-fungsi apa saja yang ada dalam sebuah sistem dan siapa saja yang menggunakan fungsi-fungsi tersebut. Dalam pembuatan sebuah *use case diagram* yang perlu ditekankan adalah "apa" yang diperbuat sistem, bukan "bagaimana" sebuah sistem berjalan. Dalam pendefinisiaan sebuah nama juga perlu sesimpel mungkin namun tetap mudah 12 dipahami agar pembaca merasa nyaman ketika membaca *use case diagram* (Sayekti, 2013). Simbol diagram *use case* dapat dilihat pada Tabel 2.1.

Tabel 2.1. Simbol diagram *use case*

No.	Gambar	Nama	Keterangan
1.	. 	*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2.	. 	*Dependency*	UHubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
3.	. 	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4.	. 	*Include*	Menspesifikasikan bahwa *use case* sumber secara eksplisit.

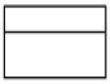
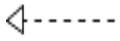
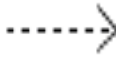
Tabel 2.1 Simbol diagram *use case* (Tabel lanjutan...)

No.	Gambar	Nama	Keterangan
5.	. 	*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6.	. 	*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7.	. 	*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8.	. 	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
9.	. 	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya.
10.	. 	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

2.10.2 *Class Diagram*

Class diagram merupakan diagram paling umum yang dijumpai dalam pemodelan berbasis UML. Di dalam *class diagram* terdapat *class* dan *interface* beserta atribut-atribut dan operasinya, relasi yang terjadi antar objek, constraint terhadap objek-objek yang saling berhubungan dan inheritance untuk organisasi *class* yang lebih baik. *Class diagram* juga terdapat *static view* dari elemen pembangun sistem. Pada intinya *class diagram* mampu membantu proses pembuatan sistem dengan memanfaatkan konsep *forward* ataupun *reverse engineering* (Sayekti, 2013). Simbol *class diagram* dapat dilihat pada Tabel 2.2.

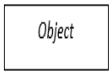
Tabel 2.2. Simbol *class diagram*

No.	Gambar	Nama	Keterangan
1.	.	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2.	. 	*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3.	. 	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4.	. 	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5.	. 	*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6.	. 	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
7.	.	*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

2.10.3 *Sequence Diagram*

*Sequence diagra*m menggambarkan interkasi antar objek di dalam dan di sekitar sistem (pengguna, *display*, dan sebagainya) berupa message yang digambarkan terhadap waktu.x *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan (Sayekti, 2013). Simbol *sequence diagram* dapat dilihat pada Tabel 2.3.

Tabel 2.3. Simbol *sequence diagram*

No.	Gambar	Nama	Keterangan
1.	. 	*Object*	*Object* merupakan *instance* dari sebuah *class* dan dituliskan tersusun secara horizontal. Digambarkan sebagai sebuah *class* (kotak) dengan nama obyek didalamnya yang diawali dengan sebuah titik koma.
2.	. 	*Actor*	*Actor* dapat berkomunikasi atau berinteraksi dengan sistem.
3.	.	*Lifeline*	*Lifeline* mengindikasikan keberadaan sebuah *object* dalam basis waktu. Notasi untuk *Lifeline* adalah garis putus-putus *vertical* yang ditarik dari sebuah obyek.
4.	. 	*Activation*	*Activation* dinotasikan sebagai sebuah kotak segi empat yang digambar pada sebuah *lifelin*e. *Activation* mengindikasikan sebuah obyek yang akan melakukan sebuah aksi.
5.	. 	*Message*	*Message*, digambarkan dengan anak panah horizontal antara *Activation*. Message mengindikasikan komunikasi antara *object-object*.

2.10.4 Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas atau *activity diagra*m menggambarkan aliran fungsionalitas sistem. Pada tahap pemodelan bisnis, diagrama aktivitas dapat digunakan untuk menujukkan aliran kerja bisnis (*business work flow*). Dapat juga digunakan untuk menggambarkan aliran kejadian (*flow of event*) dalam *use case*. Simbol activity diagram dapat dilihat pada Tabel 2.4.

Tabel 2.4. Simbol *activity diagram*

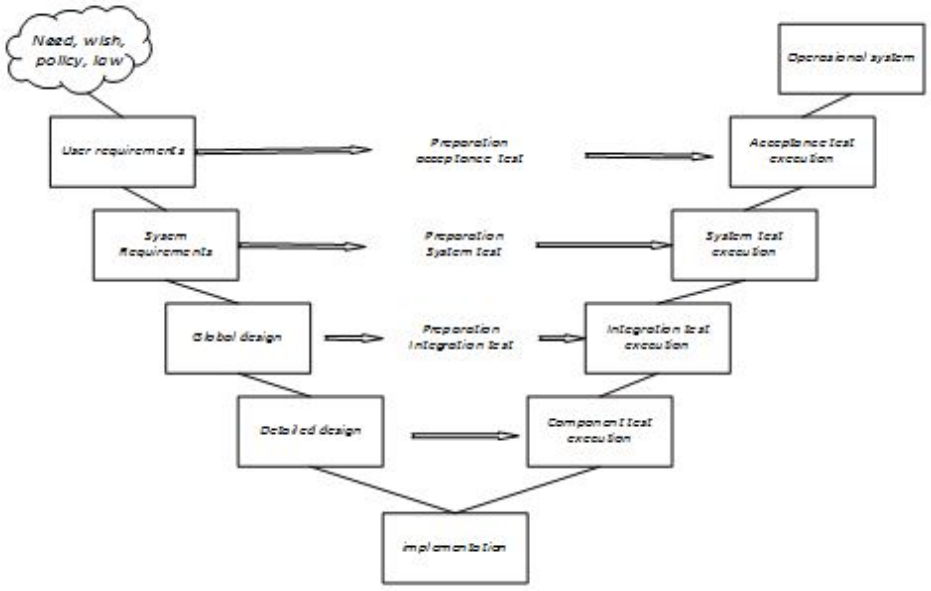
No.	Gambar	Nama	Keterangan
1.	. 	*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.

Tabel 2.4 Simbol *activity diagram* (Tabel lanjutan...)

No.	Gambar	Nama	Keterangan
2.	. 	*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi.
3.	. 	*Initial Node*	Bagaimana objek dibentuk atau diawali.
4.	. 	*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan.
5.	. 	*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.

2.11 Metode Pengembangan Sistem Menggunakan *V-Model*

V-model merupakan perluasan dari metode *waterfall*. Disebut sebagai perluasan karena tahap-tahapnya mirip dengan yang terdapat dalam model *waterfall*. Jika dalam model *waterfall* proses dijalankan secara linear, maka dalam *v-model* proses dilakukan bercabang. Dalam V-model ini digambarkan hubungan antara tahap pengembangan *software* dengan tahap pengujiannya yang dapat dilihat pada Gambar 2.2 (Graham, van Veenendaal, Evans, dan Black, 2006).

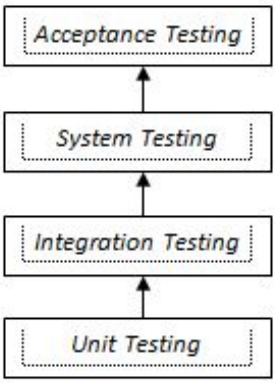


Gambar 2.2. Tahapan metode *V-Model*

Berikut penjelasan masing-masing tahap beserta tahap pengujiannya:

1. Berikut penjelasan masing-masing tahap beserta tahap pengujiannya:
   Tahap *Requirement Analysis* sama seperti yang terdapat dalam model *waterfall*. Keluaran dari tahap ini adalah dokumentasi kebutuhan pengguna. *Acceptance Testing* merupakan tahap yang akan mengkaji apakah dokumentasi yang dihasilkan tersebut dapat diterima oleh para pengguna atau tidak.
2. *Spesification* dan *System Testing*
   Dalam tahap ini analis sistem mulai merancang sistem dengan mengacu pada dokumentasi kebutuhan pengguna yang sudah dibuat pada tahap sebelumnya. Keluaran dari tahap ini adalah spesfikasi *software* yang meliputi organisasi sistem secara umum, struktur data, dan yang lain.
3. *Architecture Design* dan *Integration Testing*
   Sering juga disebut *hight level design*. Dasar dari pemilihan arsitektur yang akan digunakan berdasarkan kepada beberapa hal seperti: pemakaian kembali tiap modul, ketergantungan tabel dalam basis data, hubungan antar *interface*, detail teknologi yang dipakai. Penyajian pada tahap desainpun berbagai macam, seperti menggunakan UML dan diagram.
4. *Implementation*
   Perancangan dipecah menjadi modul-modul yang lebih kecil. Setiap modul tersebut diberi penjelasan yang cukup untuk memudahkan programmer melakukan coding. Tahap ini menghasilkan spesifikasi program seperti: fungsi dan logika tiap modul, pesan kesalahan, proses input-output untuk

tiap modul, dan lain-lain. Menurut Rifai (2015), terdapat empat fase pengujian yang dapat dlihat pada Gambar 2.3



Gambar 2.3. Tahapan fase pengujian

5. *Unit Testing*

   Unit Testing adalah proses pengujian perangkat lunak dimana masing-masing unit/komponen diuji. Tujuannya adalah untuk menvalidasi bahwa setiap unit perangkat lunak sudah melakukan seperti apa yang telah dirancang.

   Menurut Pressman (2005), *unit testing* berfokus pada upaya verifikasi terhadap unit terkecil dari perancangan perangkat lunak. Pengujian unit berfokus pada logika pemrosesan internal dan struktur data didalam komponen. *Unit testing* merupakan proses dimana pengujian dilakukan pada bagian basic dari kode program. Contohnya adalah memeriksa kode program pada *event*, *procedure*, dan *function*. *Unit testing* meyakinkan bahwa masing-masing unit tersebut berjalan sebagaimana mestinya.

6. *Integration Testing*

   Integration Testing adalah proses pengujian perangkat lunak dimana unit individu digabungkan dan diuji sebagai sebuah kelompok. Sehingga pengujian ini mampu menampilkan kesalahan dalam interaksi antar unit. Menurut Pressman (2005), pengujian integrasi adalah teknik untuk membangun arsitektur perangkat lunak, sementara pada saat yang sama melakukan pengujian untuk menemukan kesalahan terkait antarmuka. Tujuannya adalah untuk mengambil komponen yang diuji dan membangun struktur program yang telah ditentukan oleh perancangan.

   Setelah melakukan unit/ *component testing*, langkah berikutnya adalah memeriksa bagaimana unit-unit tersebut bekerja sebagai suatu kombinasi, bukan lagi sebagai suatu uni yang individual. Sebagai contoh, kita memili-

ki sebuah proses yang dikerjakan oleh dua *function*, di mana satu *function* menggunakan hasil *output* dari *function* yang lainnya. Kedua *function* ini telah berjalan dengan baik secara individu pada *unit testing*.

Pada tahap *integration testing*, kita memeriksa hasil dari interaksi kedua *function* tersebut, apakah bekerja sesuai dengan hasil yang diharapkan. Kita juga harus memastikan bahwa seluruh kondisi yang mungkin terjadi dari hasil interaksi antar unit tersebut menghasilkan *output* yang diharapkan.

7. *System Testing*

   System Testing adalah proses pengujian dimana perangkat lunak yang diuji sudah lengkap dan terintegrasi. Tujuan dari pengujian ini adalah untuk mengevaluasi kesesuaian sistem dengan persyaratan yang telah ditentukan.

8. *Acceptance Testing*

   Acceptance Testing atau uji penerimaan adalah pengujian formal dilakukan untuk menentukan apakah sistem menerima kriteria penerimaan dan memastikan jika pengguna dapat menerima sistem. Tujuan dari pengujian ini adalah untuk mengetahui tingkat kelayakan dari perangkat lunak. Seperti *integration testing*, *acceptance testing* juga meliputi pengujian keseluruhan aplikasi. Perbedaannya terletak pada siapa yang melakukan *testing*. Pada tahap ini, *end-user* yang terpilih melakukan *testing* terhadap fungsi-fungsi aplikasi dan melaporkan permasalahan yang ditemukan. Proses ini merupakan salah satu tahap final sebelum pengguna menyetujui dan menerima penerapan sistem aplikasi yang baru.

2.12 *Blackbox Testing*

Blackbox testing merupakan suatu teknik pengujian perangkat lunak dengan berfokus pada persyaratan fungsional. *Blackbox testing* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program (Pressman, 2005).

Blackbox testing berusaha menemukan kesalahan dalam kategori sebagai berikut:

1. Fungsi-fungsi yang tidak benar atau hilang.
2. Kesalahan *interface*.
3. Kesalahan dalam struktur data atau akses *database* eksternal.
4. Kesalahan kinerja.
5. Inisialisasi dan kesalahan terminasi.

Blackbox testing diaplikasikan selama tahap akhir pengujian, karena *blackbox* memperhatikan stuktur kontrol, maka perhatian berfokus pada informasi.

2.13 *User Acceptance Test* (*UAT*)

User acceptance test (UAT) merupakan sebuah pengujian sistem yang ditunjukan kepada pengguna sistem tersebut, dengan menggunakan beberapa scenario pengujian setelah aplikasi selesai dibuat. UAT umumnya dilakukan oleh klien atau pengguna akhir, biasanya tidak fokus pada identifikasi masalah sederhana seperti kesalahan ejaan, maupun di cacat *show stopper*, seperti *crash* perangkat lunak. Penguji dan pengembang mengidentifikasi dan memperbaiki masalah ini selama tahap awal pengujian fungsionalitas, pengujian saat integrasi dan pada tahap sistem testing jenis UAT yang digunakan dalam proyek akhir Komputerisasi Akuntansi adalah *blackbox testing*. UAT sering dikategorikan sebagai pengujian fungsional, sampai batas tertentu, dilihat sebagai jenis *user acceptance test*. Pada dasarnya, metode pengujian ini menganalisis fungsi tertentu tanpa membiarkan *tester* melihat struktur kode internal perangkat lunak. Oleh karena itu, *blackbox testing* juga dapat diterapkan untuk UAT, karena *blackbox testing* memiliki prinsip yang sama seperti UAT. Selama *blackbox testing*, pengguna tidak mengetahui adanya basis kode, tapi hanya tentang persyaratan yang perangkat lunak harus memenuhi. UAT untuk berorientasi objek diambil dari *use case* diagram (untuk generalisasi: cukup menerangkan *child*/anak/spesifikasi nya bukan *parent*/induk/general nya) sedangkan untuk terstruktur diambil dari DFD level terkecil.

2.14 Penelitian yang Terkait

Penelitian yang terkait adalah penelitian yang berhubungan dengan penelitian penulis saat ini, beberapa penelitian terdahulu dapat dilihat pada Tabel 2.5 Berikut:

Tabel 2.5. Penelitian terdahulu

No.	Nama Peneliti	Judul Penelitian	Hasil Penelitian
1.	Solihin (2017)	Perancangan Sistem Informasi Penerimaan Siswa Baru Berbasis *Web* (Studi Kasus: SMP Plus Babussalam Bandung)	Sistem informasi penerimaan siswa baru berbasis *web* dari hasil penelitian ini dapat memberi kemudahan akses informasi dan proses pendaftaran bagi calon siswa, mengatasi pengolahan data calon siswa menjadi lebih baik karena disimpan dalam suatu basis data yang terintegrasi.
2.	Sadikin dan Rusmawan (2017)	Sistem Pengolahan Data Penerimaan Siswa Baru dan Pembayaran Spp Pada SMK Karya Guna 1 Bekasi	Sistem pengolahan data penerimaan siswa baru dan pembayaran SPP pada SMK Karya Guna 1 Bekasi mampu memudahkan dalam proses penginputan data calon siswa baru dengan *form* pendaftaran dan *form* kelengkapan administrasi.

Tabel 2.5 Penelitian terdahulu (Tabel lanjutan...)

No.	Nama Peneliti	Judul Penelitian	Hasil Penelitian
3.	Eviani, Rizk, dan Pratiwi (2017)	Sistem Informasi Penerimaan Siswa Baru Berbasis *Web* Pada SMPN 34 Kabupaten Tebo	Sistem informasi penerimaan siswa baru membantu para calon siswa yang akan melakukan pendaftaran yaitu dengan melakukan pendaftaran secara *online* tanpa harus datang ke sekolah, membantu tugas tata usaha dalam mengolah setiap data pendaftar yang masuk atau yang melakukan pendaftaran.
4.	Nugroho dan Hidayat (2015)	Perancangan Sistem Informasi Penerimaan Siswa Baru Berbasis *Web* (Studi Kasus Di SMA Nusaputera Semarang)	Sistem informasi penerimaan siswa baru berbasis *web* menjadi sarana baru dalam penerimaan siswa baru, membantu tugas tata usaha dalam mengolah setiap data calon siswa yang masuk, membantu para calon siswa yang ingin mendaftar secara *online* tanpa harus datang ke sekolah.
5.	Astuti, Khairina, dan Febriani (2016)	Sistem Informasi Penerimaan Siswa Baru Sekolah Menengah Pertama Berbasis *Web* (Studi Kasus Kabupaten Kutai Kartanegara)	Aplikasi ini memiliki fungsi dasar antara lain pendaftaran siswa, perubahan pilihan sekolah, pengolahan data sekolah, dan cetak yang diharapkan dapat mengatasi hambatan-hambatan yang dialami baik oleh panitia penyelenggara penerimaan siswa baru maupun orang tua siswa.
6.	Farkhatin (2015)	Perancangan Sistem Informasi Penerimaan Siswa Baru	Pada sistem ini, semua kegiatan yang berhubungan dengan pendaftaran siswa baru, pengolahan nilai, absensi, jadwal pelajaran, dan laporan keuangan dilakukan tidak lagi secara manual tetapi dengan menggunakan media berupa komputer. Sistem Penerimaan Siswa Baru yang berbasis komputer dapat menangani proses Penerimaan Siswa Baru dengan cepat dan akurat serta dapat di *update* dengan mudah.
7.	Suhendar (2015)	Sistem Informasi Penerimaan Siswa Baru Berbasis *Web* (Studi Kasus Pada SMK Ciledug Al-Musaddadiyah Garut)	Terwujudnya proses pendaftaran siswa baru yang dapat diakses di luar lingkungan sekolah SMK Ciledug melalui jaringan internet, Terpenuhinya kebutuhan pengguna yang diperlukan untuk proses PSB di SMK Ciledug, Terwujudnya rancangan *database* yang berhubungan dengan proses pendaftaran calon siswa baru dan proses pengumuman hasil tes tertulis di SMK Ciledug.

Tabel 2.5 Penelitian terdahulu (Tabel lanjutan...)

No.	Nama Peneliti	Judul Penelitian	Hasil Penelitian
8.	Amin (2017)	Rancang Bangun Sistem Informasi Penerimaan Siswa Baru Pada SMK Budhi Warman 1 Jakarta	Sistem Informasi penerimaan peserta didik baru berbasis *web* pada SMK Budhi Warman 1 Jakarta dapat diakses dari mana saja kapan saja melalui jaringan internet sehingga relatif memudahkan calon siswa untuk memilih program keahlian/ kejuruan sesuai dengan keinginan dan tidak perlu datang langsung ke lokasi sekolah.
9.	Prasetyo, Cahyana, dan Himawan (2015)	Aplikasi Penerimaan Siswa Baru Berbasis *Web* (SMK Negeri 3 Yogyakarta)	Untuk meningkatkan kualitas pendidikan dan mutu penjaringan calon siswa baru, sehingga manajamen sekolah dapat melakukan pembaharuan didalam ujian masuk. Tujuan pembuatan *website* ini adalah untuk memudahkan orang tua dan siswa dalam melakukan pendaftaran tanpa harus datang ke sekolah.
10.	Nasser, Saputra, dan Syarif (2017)	Rancang Bangun Sistem Informasi Penerimaan Siswa Baru Pada SMK Negeri 7 Palopo	Berdasarkan hasil pengujian program dengan menggunakan pengujian *blackbox* yang menghasilkan tanpa kesalahan logika sehingga dapat ditarik kesimpulan bahwa sistem tersebut sudah benar. Sehingga dengan begitu sistem informasi penerimaan siswa baru SMKN 7 Palopo ini telah siap untuk diterapkan dan diimplementasikan. Dengan demikian setelah diterapkannya sistem ini maka calon siswa dapat melakukan pendaftaran dimana saja yang memiliki akses internet.