# BAB III

# METODOLOGI DAN PENELITIAN

## 3.1 Metode Penelitian

Metodologi penelitian yang digunakan dalam penelitian ini menggunakan pendekatan kualitatif atau dapat disebut dengan penelitian kualitatif. Penelitian kualitatif dapat diartikan sebagai metode penelitian yang berlandaskan pada filsafat postpositivisme/enterpretif, digunakan untuk meneliti pada kondisi obyek yang alamiah, (sebagai lawannnya adalah eksperimen) dimana peneliti sebagai instrumen kunci, teknik pengumpulan data dilakukan secara trianggulasi (gabungan), analisis data bersifat induktif/kualitatif, dan hasil penelitian kualitatif lebih menekankan makna dari pada generalisasi. (Sugiyono, 2013:38)

## 3.2 Waktu dan Tempat

Lokasi penelitian dilakukan pada Sekolah Menengah Kejuruan (SMK) Negeri 3 Kayuagung, Jalan Letnan Sayuti Kelurahan Kedaton Kec. Kota Kayuagung Kabupaten Ogan Komering Ilir Provinsi Sumatera Selatan.

## 3.3 Alat dan Bahan

Dalam mengembangkan sistem informasi penulis menggunakan perangkat keras dan perangkat lunak sebagai berikut:

### 3.3.1 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang digunakan dalam mengembangkan sistem informasi akademik berbasis web di SMK Negeri 3 Kayu Agung Kabupaten OKI adalah:

a. Notebook Acer, spesifikasi yaitu processor intel (R) Atom (R), CPU N280
b. Monitor, spesifikasi yaitu layar 11.0 inc.
c. Ram yang digunakan yaitu 2 GB.
d. Hardisk yang digunakan yaitu 320 GB.

### 3.3.2 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang digunakan dalam mengembangkan sistem informasi akademik berbasis web di SMK Negeri 3 Kayu Agung Kabupaten OKI adalah:

a. Xamp versi 3.2.2 mencakup *web server (apache), database (mysql), database manager (PhpMyadmin)*
b. Bahasa pemograman PHP
c. *Database* MySQL
d. *Web browser Mozilla Firefox / Chrome*
e. *Web* editor *Adove Dreamweaver CS6*
f. *Pencil* sebagai pembuatan desain tampilan
g. Astah sebagai alat bantu perancangan UML (*Unified Modelling Language*)
h. *Microsoft Office Visio* sebagai alat bantu pembuatan *Flowchart.*

## 3.4 Metode Pengumpulan Data

Sebagai bahan pendukung untuk mencari dan mengumpulkan data yang diperlukan dalam penelitian ini. Data yang dicari harus sesuai dengan tujuan peneliti. Beberapa metode yang digunakan yaitu :

a. Observasi (Pengamatan)

Pengumpulan data dengan menggunakan teknik observasi memiliki beberapa petunjuk yaitu :

1. Tentukanlah hal-hal apa saja yang akan dioservasi agar kegiatan observasi menghasilkan sesuai dengan yang diharapkan.
2. Mintalah ijin kepada orang yang berwenang pada bagian yang akan diobservasi.
3. Berusaha sesedikit mungkin agar tidak mengganggu pekerjaan orang lain.
4. Jika ada yang Anda tidak mengerti, cobalah bertanya. Jangan membuat asumsi sendiri. (Rosa & Shalahuddin, 2016)

b. Teknik Wawancara

Pengumpulan data dengan menggunakan wawancara memiliki beberapa petunjuk yaitu :

1. Buatlah jadwal wawancara dengan narasumber dan beritahukan maksud dan tujuan wawancara.
2. Buatlah panduan wawancara yang akan anda jadikan arahan agar pertanyaan dapat fokus kepada hal-hal yang dibutuhkan.
3. Anda boleh berimprovisasi dengan mencoba menggali bagian-bagian tertentu yang menurut anda penting.
4. Catat hasil wawancara tersebut.(Rosa & Shalahuddin, 2016)

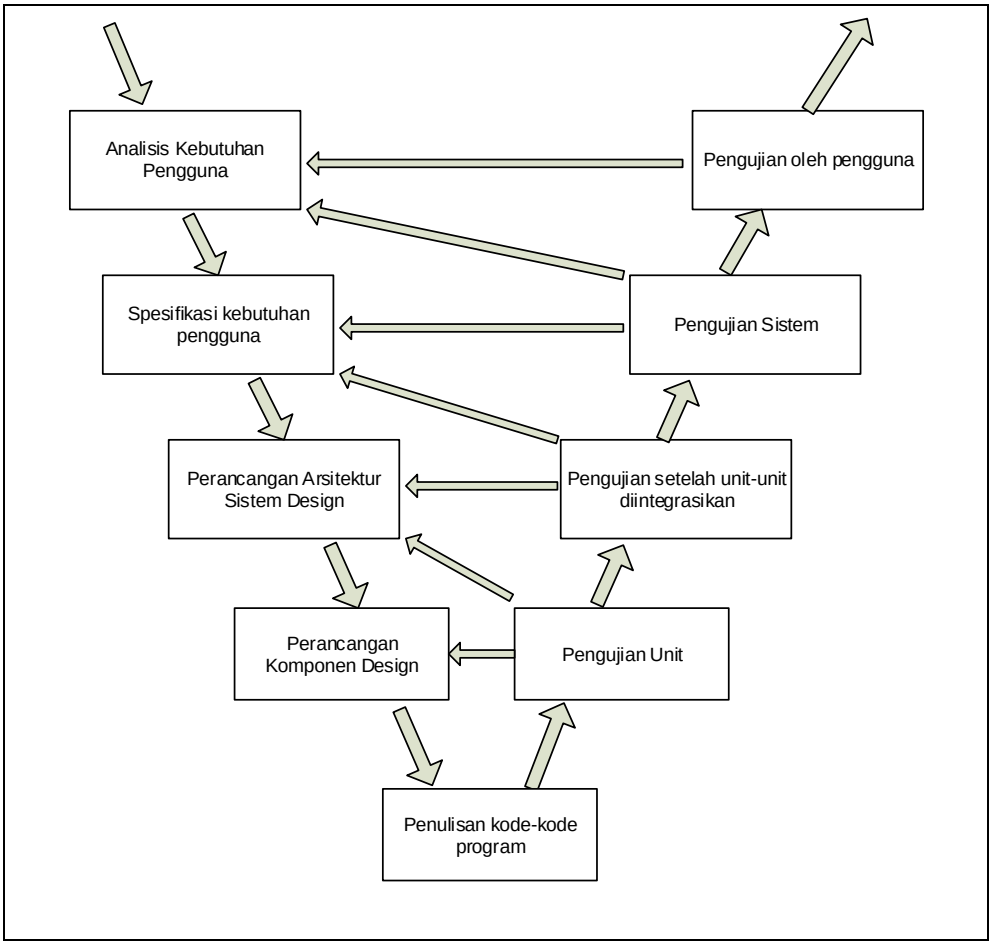
c. Studi Pustaka

Studi kepustakaan merupakan langkah yang penting dimana setelah seorang peneliti menetapkan topik penelitian, langkah selanjutnya adalah melakukan kajian yang berkaitan dengan teori yang berkaitan dengan topik penelitian. Dalam pencarian teori, peneliti akan mengumpulkan informasi sebanyak-banyaknya dari kepustakaan yang berhubungan. Sumber-sumber kepustakaan dapat diperoleh dari : buku, jurnal, majalah, hasil-hasil penelitian (tesis dan disertasi), dan sumber-sumber lainnya yang sesuai (internet, koran dll). (Nazir, 1998)Penulis juga menggunakan metode pengumpulan data dengan studi pustaka yang dilakukan oleh penulis yaitu mencari data secara langsung dari sumber-sumber lain seperti buku, jurnal dan hasil penelitian yang berkaitan dengan permasalahan.

## 3.5 Metode Pengembangan Sistem

Model-*V* ini merupakan perluasan dari model *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. Disebut bercabang yaitu jika ditahap pengujian terjadi pesan error atau tidak sesuai dengan perancangan pengembangan software maka akan di koreksi ditahap sebelumnya begitupun dengan tahap selanjutanya. Dalam V-Model ini digambarkan hubungan antara tahap pengembangan *software* dengan tahap pengujiannya (Anonim, 2016). Sedangkan menurut (Munnasar, 2010) Tahapan V-Model hampir sama dengan *waterfall*,

hanya pada model ini tahapan pengujian dirinci untuk masing-masing tahap pada Gambar. 3.1




**Gambar. 3.1** Model-***V***

Sumber : Pressman., *"Rekayasa Perangkat Lunak – Pendekatan Praktisi Edisi 7 (Buku 2)*"., 2017, hal. 47.

Berikut ini penjelasan mengenai tahapan pada metode pengembangan yang digunakan (Windi Eka Y.R, Saiful Bukhori, D. I. 2013), yaitu:

1. Analisis kebutuhan pengguna & Pengujian oleh pengguna

   Tahap analisis kebutuhan pengguna sama seperti yang terdapat dalam model *waterfall*. Keluaran dari tahap ini adalah dokumentasi kebutuhan pengguna.

Pengujian oleh pengguna merupakan tahap yang akan mengkaji apakah dokumentasi yang dihasilkan tersebut dapat diterima oleh para pengguna atau tidak.

2. Spesifikasi kebutuhan pengguna & Pengujian sistem keseluruhan

   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 spesifikasi software yang meliputi organisasi sistem secara umum, struktur data, dan yang lain.

   Pengujian sistem keseluruhan adalah pada tahap ini akan dilakukan pengujian dengan mencocokan sistem dan kebutuhan yang diinginkan user

3. Perancangan arsitektur design & Pengujian unit integrasi.

   Sering juga disebut *High Level Design*. Dasar dari pemilihan arsitektur yang akan digunakan berdasar kepada beberapa hal seperti: pemakaian kembali tiap modul, ketergantungan tabel dalam basis data, hubungan antar *interface*, detail teknologi yang dipakai.

   Pengujian unit integrasi merupakan pengujian pada tahap ini dilakukan oleh pengembang pada seluruh unit sistem yang berkaitan dengan *input* dan *output* sistem. Jika ditemukan kesalahan, maka akan dilakukan pengujian lagi terhadap seluruh sistem.

4. Perancangan komponen design & Pengujian unit.

   Sering juga disebut sebagai *Low Level Design*. 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.

Pengujian unit ialah pengujian dilakukan pada semua unit untuk menguji alur sistem minimal satu kali.

5. Coding

   Dalam tahap ini dilakukan pemrograman terhadap setiap modul yang sudah dibentuk.

## 3.6 Pengujian Sistem

Pengujian perangkat lunak sebenarnya merupakan sebuah proses verifikasi dan validasi. Verifikasi adalah tahapan dari rekayasa perangkat lunak untuk memastikan produk yang di hasilkan dari aktivitas pengembangan sesuai dengan spesifikasi yang di tentukan. Sedangkan validasi merupakan tahapan untuk memberikan penilaian produk sesuai dan memuaskan keinginan dari pemangku kepentingan (Pressman, 2010 : 550). Sedangkan menurut (Rosa dan Shalahuddin, 2016 : 272) Pengujian adalah satu set aktifitas yang direncanakan dan sistematis untuk menguji atau mengevaluasi kebenaran yang diinginkan.

Proses pengujian perangkat lunak dilakukan untuk memastikan bahwa perangkat lunak yang dikembangkan sudah berjalan dengan semestinya. Pengujian perangkat lunak dilakukan dengan 4 tahap dapat dilihat pada Tabel 3.1 berikut;

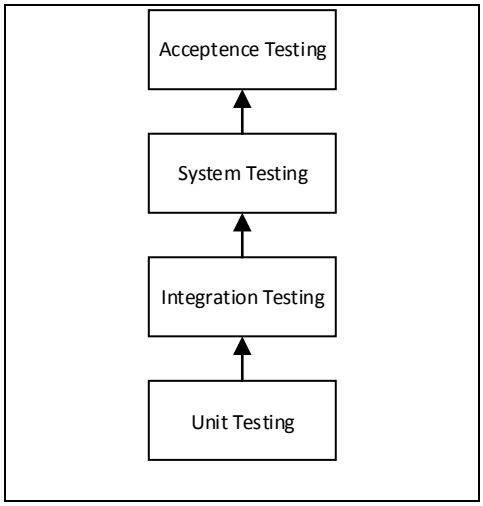
**Tabel 3.1** Metode Pengujian

| **Teori Pressman** | **Model-V** | **Teknik Pengujian** | **Aspek Uji** |
|---|---|---|---|
| Verifikasi | *Unit Testing* | *White Box* | |
| | *Integration Testing* | *Black Box* | *Functionality* |
| Validasi | *System Testing* | *Instability Test* | *Compatibility* |
| | *Acceptence Testing* | *Playability Test* | *Playability* |

Pada Tabel 3.1 merupakan tahapan pengujian pada metode pengembangan sistem menurut teori Pressmen dan dihubungankan dengan metode pengembangan sistem Model-V, yang mana Verifikasi bagian dari pengujian *Unit Testing* dan *Integration Testing* teknik pengujian yang di pakai yaitu *white box* dan *black box* serta aspek yang diuji *Functionality*. Kemudian Validasi merupakan bagian dari pengujian *System Testing* dan *Acceptence Testing*, teknik pengujian yang dipakai *Instability Test* dan *Playability Test*, aspek yang diuji *Compability* dan *Playability*.

### 3.6.1 Fase Pengujian

Pada Gambar 3.2 akan menggambarkan fase pengujian yang ada pada metode pengembangan sistem Model-V, berikut gambar fase pengujian;




**Gambar 3.2** Fase Pengujian Model-*V*

Terdapat empat fase pengujian didalam metode pengembangan sistem Model-*V* yaitu;

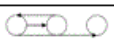
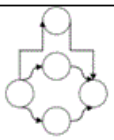
1. *Unit Testing*

   *Unit testing* adalah proses pengujian perangkat lunak dimana masing-masing unit/komponen diuji. Tujuannya untuk meverifikasi bahwa setiap unit perangkat lunak sudah melakukan seperti apa yang telah dirancang. Menurut Pressmen, *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* dapat dilakukan dengan menggunakan metode *whitebox testing* (*functionality*)

   Pengujian *whitebox* merupakan sebuah filosofi perancangan *test cash* yang menggunakan struktur kontrol yang dijelaskan sebagai bagian dari perancangan peringkat komponen untuk menghasilkan *test case* (Pressman, 588:2012). Pada pengujian *whitebox* memiliki tahapan pengujian untuk mendapatkan *test cash* diantaranya membuat notasi diagram alir, *cyclomatic complexity*, dan jalur independen.

   a) Notasi diagram alir

      Notasi sederhana untuk representasi aliran kontrol disebut juga diagram alir (*grafik program*). Grafik alir menggambarkan aliran kontrol logika yang menggunakan notasi alir pada gambar 3.3 ini;

| Notasi | Arti |
|---|---|
|  | Skema Sequence |
|  | Skema If |
|  | Skema While (...) DO (...) |
|  | Skema Repeat (...) Until (...) |
|  | Skema Case (...) Of |

(Sumber: Roger Pressmen, 2012)

**Gambar 3.3** Notasi grafik alir

Untuk menggambarkan *flowgraph* dapat menggunakan simbol *flowgraph* yang terbentuk lingkaran untuk mempresentasikan satu atau lebih statemen prosedural. Anak panah pada *flowgraph* disebut *edges*. *Edge* harus berhenti pada suatu simbol, meskipun bila simpul tersebut tidak mempresentasikan statement prosedural (misalnya, lihat simpul untuk bangun *if-then-else*).

b) Jalur independen

Jalur independen adalah jalur yang melalui program yang memperkenalkan setidaknya satu kumpulan pernyataan-pernyataan pemrosesan atau kondisi baru. Bila dinyatakan dengan *flowgraph* jalur independen harus bergerak sepanjang paling tidak satu *edges* yang tidak dilewatkan sebelum jalur tersebut ditentukan. Berikut contoh jalur independen;

*Path* 1 = 1-11

*Path* 2 = 1-2-3-4-5-10-1-11

*Path* 3 = 1-2-3-6-8-9-10-1-11

*Path* 4 = 1-2-3-6-7-9-10-1-11

Perhatikan bahwa masing-masing jalur harus memperkenalkan sebuah *edge* baru.

Selanjutnya menentukan fondasi *cyclomatic complexity* adalah teori grafik, dan memberikan metriks perangkat lunak yang sangan berguna. *Cyclomatic complexity* dapat dihitung dalam salah satu dati tiga cara berikut;

a) Jumlah region grafik alir sesuai dengan *cyclomatic complexity*
b) *Cyclomatic complexity* V(G) untuk grafik alir dihitung dengan rumus : ***V(G) = E – N +2***

   Dimana : E = jumlah *edge* pada grafik alir

   N = jumlah *node* pada grafik alir
c) *Cyclomatic complexity* V(G) juga dapat dihitung dengan rumus:

   ***V(G) = P + 1***

   Dimana P = jumlah *predicate node* pada grafik alir

Pada *unit testing*, pengujian dilakukan dengan dengan memeriksa bagian kode program secara terpisah dari bagian yang lain. Pengujian dapat langsung dilakukan setiap kali sebuah kode unit (*event*, *procedure*, *functon*) selesai dibuat. *Unit testing* dapat dilakukan dengan menggunakan metode *whitebox testing* (*functionality*)

2. *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, 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 unit yang individual. Sebagai contoh, kita memiliki 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. *Integration testing* dapat dilakukan dengan pengujian *blackbox* (*functionality*).

.

3. *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. Menurut Pressman, pengujian sistem merupakan serangkaian

pengujian yang berbeda-beda yang bertujuan untuk memverifikasi bahwa semua elemen sistem telah terintegrasi dengan baik dan menjalankan fungsi yang telah ditetapkan. *System Testing* mencakup pengujian aplikasi yang telah selesai dikembangkan. Karena itu, aplikasi harus terlihat dan berfungsi sebagaimana mestinya terhadap pengguna akhir.

*System Testing* dapat dilakukan menggunakan metode *Blackbox*, *nonfunctional test* (*configuration test*, *compability test*, *stress test*, *performance test* dan lain sebagainya

4. *Acceptence 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 perangakat 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. Pengujian yang dilakukan merupakan simulasi penggunaan nyata dari aplikasi pada lingkungan yang sebenarnya. Proses ini merupakan salah satu tahap final sebelum pengguna menyetujui dan menerima penerapan sistem aplikasi yang baru. Karena itu pada tahap ini sudah tidak difokuskan untuk mengangkat permasalahan kecil seperti

kesalahan pengetikan. Hal-hal minor seperti di atas sudah seharusnya ditangani selama *Unit/Component Testing* dan *Integration Testing*. *Acceptance test* dapat dilakukan menggunakan metode *Blackbox, usability* atau *playbility* .

Menurut Pressmen (2010) pengujian perangkat lunak adalah proses verifikasi dan validasi perangkat lunak yang diuji. Pada *unit testing* dan *integration testing* merupakan tahapann verifikasi pengujian perangkat lunak yang dilakukan dengan metode *whitebox* dan *blackbox*. Kemudian pada tahap validasi pengujian perangkat lunak yaitu pada *system testing* dan *acceptence testing* dilakukan pengujian alpha untuk mengukur *compability* dan *playbility* perangkat lunak.

Variabel penelitian yang ada dalam peneltian Sistem Informasi Akademik berbasis web di SMK Negeri Kayu Agung ini adalah;

1. *Functionality* (fungsionalitas) merupakan kemampuan perangkat lunak untuk menyediakan fungsi sesuai kebutuhan pengguna, ketika digunakan dalam kondisi tertentu.
2. *Compability* (kompatibel) merupakan kemampuan perangkat lunak untuk ditransfer dari satu lingkungan ke lingkungan lain.
3. *Playability* (kemampuan untuk dimainkan) merupakan kemampuan perangkat lunak untuk mudah dipahami, dipelajari, digunakan, dan menarik bagi pengguna ketika digunakan dalam kondisi tertentu.