BAB II
LANDASAN TEORI

2.1 Penelitian Terkait

Penelitian Terkait yang pernah dilakukan sebelumnya antara lain sebagai berikut.

a. Penelitian [1] berjudul Penerapan Algoritma Quick Pivot Sort dan Radix Sort Pada Data Buku. Penelitian ini membandingkan pengurutan Quick Sort dan Radix Sort pada data buku dengan cara membandingkan menggunakan program pengurutan yang dibuat serta kedua metode sorting tersebut dibandingkan menggunakan penghitungan hasil dari perbandingan tersebut adalah metode quick sort lebih efektif dalam mengurut data di bandingkan dengan Radix Sort.
b. Penelitian [2] berjudul Perbandingan Kecepatan Gabungan Algoritma Utama Quick Sort dan Merge Sort dengan Algoritma Tambahan Insertion Sort, Bubble Sort dan Selection Sort. Penelitian ini membandingkan kecepatan metode algoritma pengurutan data, hasil yang didapatkan bahwa Quick Sort adalah algoritma yang paling cepat serta untuk data skala kecil Insertion Sort dan Selection Sort yang digunakan, untuk data memiliki pola dan aturan tertentu Bubble Sort dan Insertion Sort yang digunakan, dan untuk data skala besar Quick Sort dan Merge Sort yang digunakan.
c. Penelitian [3] berjudul Perbandingan Metode Selection Sort dan Insertion Sort dalam Pengurutan Data Menggunakan Bahasa Program Java. Penelitian ini membandingkan Metode Selection Sort dan Insertion Sort dalam pengurutan data menggunakan bahasa Program Java, hasil yang didapatkan dengan data relatif sedikit (13 data) hanya membutuhkan waktu sekitar 1 detik untuk mengeksekusinya. Algoritma Seleciton Sort lebih sederhanya karena hanya membandingkan suatu data dengan data lain yang berada pada rentang posisi tertentu, dan Algoritma Insertion Sort sedikit lebih komples, karena memerlukan suatu variable untuk menampung data sementara,

2.2 *Quick Sort*

Menurut (Sabarudi Waruwu , Abdul Sani Sembiring , Matias Julyus. F. S, 2019) *Quicksort* merupakan Algoritma pengurutan yang dikembangkan oleh Tony Hoare. performa rata-rata pengurutan O(n log n) untuk mengurutkan n item. Algoritma ini juga dikenal sebagai Partition-Exchange Sort atau disebut sebagai Sorting Pergantian Pembagi. Quicksort merupakan

Algoritma Pembagi. Pertama quicksort membagi list yang besar menjadi dua buah sub list yang lebih kecil: element kecil dan element besar. Quicksort kemudian dapat menyortir sub list itu secara rekursif.

2.3 *Insertion Sort*

Menurut (Panny Agustia Rahayuningsih, 2016) Insertion Sort adalah sebuah algortima sederhana yang cukup efisien untuk mengurutkan sebuah list yang hampir terurut. Algoritma ini juga bisa digunakan sebagai bagian algoritma yang lebih canggih. Cara kerja algoritma ini adalah dengan mengambil elemen list satu-per-satu dan memasukkannya di posisi yang benar seperti namanya. Pada array, list yang baru dan elemen sisanya dapat berbagi tempat di array, meskipun cukup rumit. Untuk menghemat memori, implementasinya menggunakan pengurutan di tempat yang membandingkan elemen saat itu dengan elemen sebelumnya yang sudah diurut, lalu menukarnya terus sampai posisinya tepat. Hal ini terus dilakukan sampai tidak ada elemen tersisa di input.

2.4 *Web Mobile*

Menurut (Saipul Anwar, Yasin Efendi, Rushendra Rustam, Andrew, 2016) Web Mobile bertujuan untuk mengakses layanan data secara wireless dengan menggunakan perangkat mobile seperti Telepon Genggam, pada dan perangkat portable yang tersambung ke sebuah jaringan telekomunikasi selular. Mobile web yang diakses melalui perangkat mobile perlu dirancang dengan mempertimbangkan keterbatasan perangkat mobile seperti sebuah handphone yang memiliki sebuah layar dengan ukuran yang terbatas ataupun beberapa keterbatasan pada sebuah perangkat mobile.

2.5 *Mobile*

Menurut Suryadi, 2013, Dalam (Hafiz Irsyad, 2016) Secara bahasa, istilah c sendiri dapat diartikan sebagai sesuatu yang bergerak, sesuatu yang mudah dibawa kemana-mana. Dan di sini kita akan langsung batasi pengertian dari perangkat mobile sebagai alat untuk komunikasi. Jadi, dengan adanya perangkat mobile (mobile device), dimanapun kita berada, kapan pun waktunya, dan apa pun aktifitasnya, kita akan dapat dengan mudah melakukan hubungan komunikasi

dengan siapapun. “Dengan perangkat mobile, dunia dalam genggaman”. Setidaknya kalimat tersebut cukup mewakili pengertian dari perangkat mobile secara umum.

2.6 *Android*

Menurut (Hetin Tandi Arru1, Arif Harjanto, 2018) Android merupakan suatu software (perangkat lunak) yang digunakan pada mobile device (perangkat berjalan) yang meliputi sistem operasi, middleware dan aplikasi inti”. Android adalah sebuah sistem operasi untuk smartphone dan tablet. Sistem operasi dapat diilustrasikan sebagai jembatan antara piranti (device) dan penggunaannya, sehingga pengguna dapat berinteraksi dengan device-nya dan menjalankan aplikasi-aplikasi yang tersedia pada device. Sistem operasi Android ini bersifat open source sehingga banyak sekali programmer yang berbondong-bondong membuat aplikasi maupun memodifikasi sistem ini. Para programmer memiliki peluang yang sangat besar untuk terlibat mengembangkan aplikasi Android karena alasan open source tersebut. Sebagian besar aplikasi yang terdapat dalam Play Store bersifat gratis dan ada juga yang berbayar.

2.7 ***Prototype***

Menurut (Pressman, 2017) *Prototype* adalah proses pembuatan model sederhana *software* yang mengijinkan pengguna memiliki gambaran dasar tentang program serta melakukaan pengujian awal. *Prototype* memberikan fasilitas bagi pengembang dan pengguna untuk saling berinteraksi selama proses pembuatan, sehingga pengembang dapat dengan mudah memodelkan perangkat lunak yang akan di buat. Metode ini cocok digunakan untuk mengembangkan sebuah perangkat lunak yang dikembangkan kembali.Metode ini dimulai dengan pengumpulan kebutuhan pengguna. Kemudian membuat sebuah rancangan kilat yang selanjutnya akan dievaluasi kembali sbelum di produksi secara benar *Prototype,* bukanlah merupakan sesuatu yang lengkap, tetapi sesuatu yang harus dievaluasi dan dimodifikasi kembali.Segala perubahan dapat terjadi pada saat *prototype* dibuat untuk memenuhi kebutuhan pengguna dan saat yang sama memungkinkan pengembangan untuklebih memahami kebutuhan pengguna secara baik.

Tahapan-tahapan dalam metode *Prototype* :

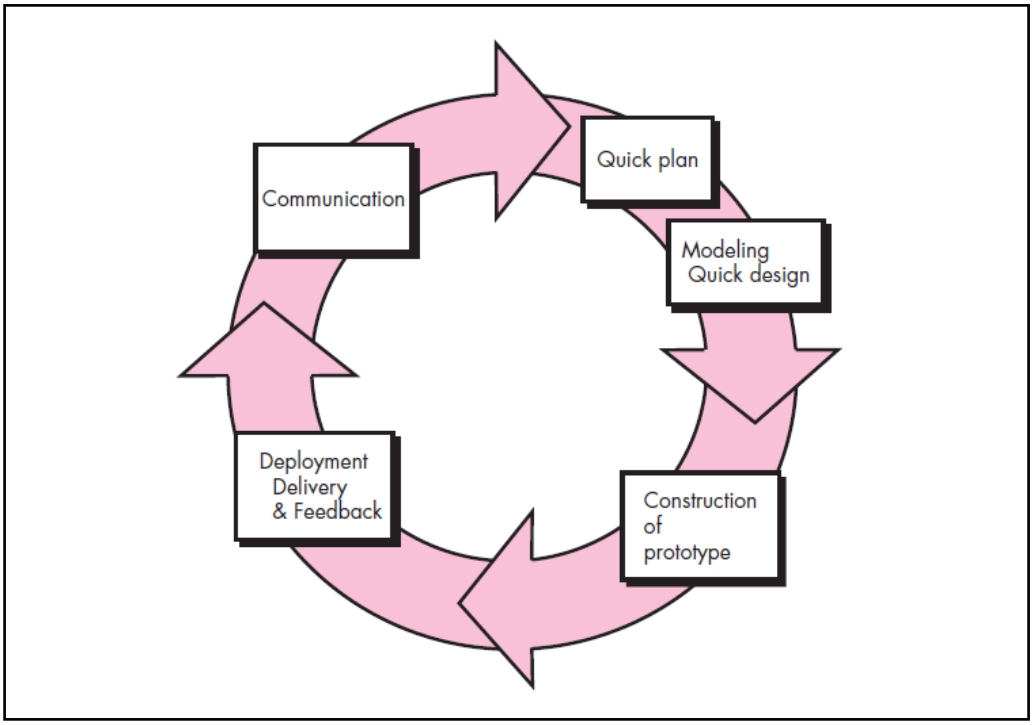
1. Komunikasi (*Communication*) :pengumpulan data awal, yaitu komunikasi dengan klien dan *user* untuk menentukan kebutuhan.

2. Perencanaan Cepat (*Quick Plan*) : pembuaan perencanaan analisis terhadap kebutuhan pengguna.

3. Pemodelan Perancangan Cepat (*Modeling Quick Design*) : membuat rancangan desain program.

4. Pembentukkan S*Prototype* (*Construction of prototype*) : pembuatan aplikasi berdasarkan dari pemodelan desain yang telah dibuat.

5. Penyerahan Sistem dan Umpan Balik (*Development Delevery and Feedback*) :memproduksi perangkat ssecara benar sehinga dapat digunakan oleh pengguna.



Gambar 2.1 Diagram Prototype

Pada Gambar 2.1, Tahap pertama ialah *communication* dan pengumpulan data awal yaitu tahap suatu perencanaanyang di lakukan, mulai dari menciptakan dan melaksanakan proses untuk memastikan bahwa perencanaan tersebut berkualitas tinggi, terpercaya, efisiensi biaya.Tahap kedua adalah *quick plan* yaitu analisis terhadap kebutuhan pengguna.Tahap ketiga adalah *modelling quick design* yaitu pembuatan desain secara umum unuk selanjutnya dikembangkan kembali.Tahap keempat adalah *construction of prototype* adalah pembuatan

perangkat*prototype* termasuk pengujian dan penyempurnaan.Tahap kelima adalah *deployment, delivery, and feedback* adalah tahap penyerahan sistem ke pengguna dan umpan balik.

2.8 PHP

Menurut Budi Raharjo, 2016, Dalam (Halimah, Bobby Bachry, 2018) Salah satu bahasa pemrograman skrip yang dirancang untuk membangun suatu aplikasi web.

2.9 *MySQL*

Menurut (Achmad Solichin, 2014) MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: database management system) atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL.

2.10 *Use Case* Diagram

Menurut (Ade Hendini, 2016) Use case diagram merupakan pemodelan untuk kelakuakn (behavior) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam Use Case Diagram yaitu:

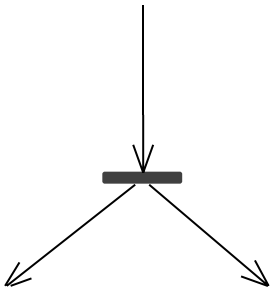
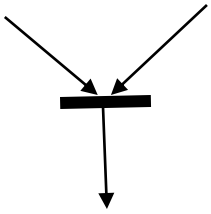
Gambar	Keterangan

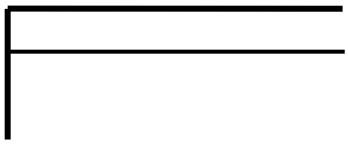
	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktir, yang dinyatakan dengan menggunakan kata kerja
	*Actor* atau Aktor adalah *Abstraction* dari orang atau sistemyang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktir, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan*Use Case*, tetapi tidak memiliki kontrol terhadap use case.
	Asosiasi antara aktor dan use case, digambarkan dengan garis tanpa panah yang mengindikasika n siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan data.

	Asosiasi antara aktor dan use case yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.
- - - - - - - - - - - - <<include>>	Include, merupakan di dalam use case lain (required) atau pemanggilan use case oleh use case lain, contohnya adalah pemanggilan sebuah fungsi program.
<- - - - - - - - - - - - <<extends>>	Extend, merupakan perluasan dari use case lain jika kondisi atau syarat.

2.11 Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam activity Diagram yaitu:

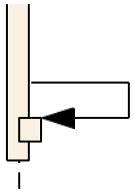
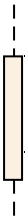
Gambar	Keterangan
	Start Point, diletakkan pada pojok kiri atas dan merupakan awal aktivitas.
	End Point, akhir aktivitas.
	Activities, menggambar kan suatu proses / kegiatan bisnis.
	Fork/percabangan, digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan parallel menjadi satu.
	Join (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	Decision Points, menggambarkan pilihan untuk pengambilan keputusan, tru atau false.

	Swimlane, pembagian activity diagram untuk menunjukkan siapa melakukan apa.

2.12 Diagram Urutan (*Sequence Diagram*)

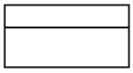
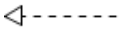
Sequence Diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang di kirimkan dan diterima antar objek symbol-simbol yang digunakan dalam Squence Diagram yaitu:

Gambar	Keterangan
	Entity Class, merupakan bagian dari system yang berisi kumpulan kelas berupa entitas - entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	Boundary Class, berisi kumpulan kelas yang menjadi interface atau interaksi antara satu atau lebih actor dengan sistem, seperti tampilan form entry dan form cetak.
	Control class, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.

	Message, simbol mengirim pesan antar class
	Recursive, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	Activation,mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivasi sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang lifeline terdapat activation.

2.13 Diagram Kelas (*Class Diagram*)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Class Diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan. Class Diagram secara khas meliputi : Kelas (Class), Relasi Assosiations, Generalitation dan Aggregation, attribut (Attributes), operasi (operation/method) dan visibility, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan Multiplicity atau Cardinality.

SIMBOL	KETERANGAN
Generalization 	Menggambarkan relasi generalisasi
Nary Association 	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek
Class 	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
Collaboration 	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
Realization 	Menggambarkan relasi.
Dependency 	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
Association	Apa yang menghubungkan antara objek satu dengan objek lainnya

2.14 Blackbox Testing

Menurut Rosa A.S & M. Shalahuddin, 2013, Dalam (Neni Purwati, Halimah, Agus Rahardi, 2018) Black box testing adalah pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian black box dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan.