RANCANG BANGUN APLIKASI INFORMASI WALISONGO BERBASIS *LOCATION BASED SERVICE* PADA *SMARTPHONE ANDROID*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:

MOHAMMAD ROZY ARDIANSYAH

NIM. 105090605111001

PROGRAM STUDI TEKNIK INFORMATIKA/ILMU KOMPUTER
JURUSAN INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG

2016

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PENGESAHAN

RANCANG BANGUN APLIKASI INFORMASI WALISONGO BERBASIS *LOCATION BASED SERVICE* PADA *SMARTPHONE ANDROID*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Mohammad Rozy Ardiansyah
NIM: 105090605111001

Skripsi ini telah diuji dan dinyatakan lulus pada
21 Juli 2016
Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	Dosen Pembimbing II
Agi Putra Kharisma, S.T., M.T.	Aswin Suharsono,S.T.,M.T.
NIK. 201304 860430 1 001	NIK. 201102 840919 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T,M.T,Ph.D
NIP. 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang 21 Juli 2016

Mohammad Rozy Ardiansyah

NIM: 105090605111001

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul " Rancang Bangun Aplikasi Informasi Walisongo Berbasis *Location Based Service* pada *Smartphone Android*" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Agi Putra Kharisma, S.T., M.T. Selaku dosen pembimbing I yang telah banyak memberikan ide, ilmu, pemikiran dan saran dalam penyusunan tugas akhir ini.

2. Bapak Aswin Suharsono, S.T., M.T. Selaku dosen pembimbing II yang telah memberikan masukan, ilmu dan saran dalam menyelesaikan skripsi ini.

3. Bapak Drs. Marji, M.T dan Bapak Issa Arwani, S.Kom, M.Sc selaku Ketua dan Sekretaris Program Studi Informatika/Ilmu Komputer Fakultas Ilmu Komputer Universitas Brawijaya.

4. Bapak Nanang Yudi Setiawan, ST selaku dosen pembimbing akademik yang telah memberikan bimbingan, ilmu dan saran selama penulis belajar.

5. Bapak Ardjoni, Ibu Riris Tuti Supriyati, Shella Puspitasari dan seluruh keluarga besar, atas segala nasehat, kasih sayang, perhatian, dukungan, dan kesabarannya dalam membesarkan dan mendidik penulis, serta senantiasa memberikan doa dan semangat dalam pengerjaan skripsi ini.

6. Seluruh Civitas Akademik Program Studi Informastika/Ilmu Komputer Fakultas Ilmu Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi dan selama penyelesaian skripsi ini.

7. Teman – Teman DR-Z yang telah memberikan semangat kepada penulis, Winarto, Adi, dan Setio.

8. Teman – Teman di gresik yang telah memberikan semangat kepada penulis, Oditio, Ritma, Gres, Wendy, Areta, Didik, dan Antok

9. Teman – Teman Angkatan 2010 Program Studi Informatika/Ilmu Komputer yang telah memberikan segala bantuannya selama menempuh studi ini.

10. Semua pihak yang telah membantu dan berbagi ilmu dalam penyelesaian skripsi, yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa penyusunan skripsi ini masih banyak kekurangan baik format penulisan maupun isinya. Oleh karena itu, kritik dan saran membangun dari pembaca dapat membantu untuk pengembangan diri. Semoga skripsi ini dapat memberikan manfaat bagi semua pihak.

Malang, 21 Juli 2016

Penulis

rozyk54@hotmail.com

ABSTRAK

Mohammad Rozy Ardiansyah.2016. Rancang Bangun Aplikasi Informasi Walisongo Berbasis *Location Based Service* pada *Smartphone Android*. Skripsi Program Studi Teknik Informatika / Ilmu Komputer, Fakultas Ilmu Komputer Universitas Brawijaya. Pembimbing: Agi Putra Kharisma,S.T.,M.T Dan Aswin Suharsono,S.T.,M.T

Ziarah Walisongo adalah kegiatan dimana peziarah mengunjungi sembilan makam pemuka agama yang menyebarkan pengetahuan tentang islam di sekitar pulau jawa. Tingginya jumlah peziarah Walisongo ini menyebabkan masalah bagaimana menentukan rute tercepat untuk menuju makam para wali. Rancang Bangun Aplikasi Informasi Sistem Walisongo Berbasis *Location Based Service* Pada *Smarthphone Android* dapat memberikan fasilitas apa yang peziarah butuhkan. Aplikasi ini menggunakan algoritma *Ant Colony Optimization* untuk mencari beberapa kemungkinan rute terbaik dengan suatu graf. Kemudian dilanjutkan dengan algoritma *Brute Force* untuk menemukan rute tercepat. Aplikasi ini diuji dengan menggunakan pengujian unit, integrasi, validasi, *usability* dan kompatibilitas. Pada saat pengujian unit dan integrasi dapat disimpulkan bahwa modul setiap sistem telah memenuhi persyaratan fungsional dari perhitungan *Cyclomatic Complexity*. Uji validasi menunjukkan bahwa aplikasi telah memenuhi kebutuhan rekayasa persyaratan. pengujian *Usability* diberikan kepada 30 responden dengan menggunakan kuesioner *System Usability Scale* (SUS). Didapatkan hasil 85% untuk kemudahan penggunaan aplikasi dan mencapai 89% untuk pengujian hasil pencarian rute tercepat. Dalam pengujian kompatibilitas menunjukkan bahwa aplikasi dapat berjalan pada semua perangkat *Android* yang menggunakan versi os 2.3 keatas

Kata kunci : *Android, Ant Colony Optimization, Brute Force, Location Service,* walisongo

ABSTRACT

Mohammad Rozy Ardiansyah.2016. Design Application Information Walisongo *Location Based Service* on *Smartphone Android*. Thesis of Computer Science, Fakultas Ilmu Komputer Universitas Brawijaya. Supervisor: Agi Putra Kharisma,S.T.,M.T & Aswin Suharsono,S.T.,M.T

The Pilgrimage of Walisongo is an activity where people visit a sacred graves of nine religious leader who spread the knowledge about Islam around Java. The high number of Walisongo's pilgrim causing a problem to determine the fastest route to reach those sacred place. The aim of this research to build an information system application of Walisongo based on Android Smartphone that can facilitate the pilgrims need. This application using Ant Colony Optimization Algorithm to calculate the best route by graph. Then continued with Brute Force Algorithm to find the fastest route. This application was tested by using unit, integration, validation, usability and compatibility. The unit and integration test concluded that each system's modul has fulfilled the functional requirement from the calculation of cyclomatic complexity. The validation test showed that applications have fulfilled the requirement engineering needs. Usability test applied to 30 respondents using The System Usability Scale (SUS) questionnaire. This application gets 85% for the ease of use dan reach 89% for the test result. The compatibility test indicates that application can runs on all Android devices which 2.3 Android OS and above.

keyword : Android, Ant Colony Optimization, Brute Force, Location Service, walisongo

Daftar Isi

PENGESAHAN i
PERNYATAAN ORISINALITAS ii
KATA PENGANTAR iii
ABSTRAK v
ABSTRACT vi
Daftar Isi vii
Daftar Gambar x
Daftar Tabel xii
BAB 1 PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan 2
1.5 Manfaat 2
1.6 Sistematika Pembahasan 3
BAB 2 LANDASAN TEORI 5
2.1 Kajian Pustaka 5
2.2 Pengertian Informasi 6
2.3 LBS (Location Based Service) 6
2.4 Graf 6
2.4.1 Jenis-Jenis Graf 7
2.4.2 Lintasan dan Sirkuit Hamilton 8
2.5 *Travelling Salesman Problem* (TSP) 8
2.6 Algoritma *Ant Colony Optimization* 9
2.6.1 Cara Kerja Semut Menemukan Rute Terpendek Dalam ACO 10
2.6.2 Penetapan Parameter Algoritma *Ant Colony Optimization* 14
2.7 Algoritma *Brute Force* 17
2.8 *Waypoint* 18
2.8.1 Perhitungan Jarak Antara Dua *Waypoints* 18
2.9 *Global Positioning System* (GPS) 19

2.9.1 Fungsi GPS 19
2.9.2 Ketelitian GPS 19
2.9.3 Cara Kerja GPS 19
2.9.4 *Assisted-Global Positioning System* (A-GPS) 20
2.10 Google Maps Android API 21
2.11 Sistem Operasi Android 21
2.11.1 Fitur Android 22
2.11.2 Arsitektur Android 22
2.11.3 Fundamental Aplikasi 25
2.12 Unified Modeling Language 26
2.12.1 Use Case Diagram 26
2.12.2 Class Diagram 26
2.12.3 Sequence Diagram 27
2.12.4 Activity Diagram 27
2.13 Konsep Teknik Dan Strategi Pengujian 28
2.13.1 Pengujian Unit 28
2.13.2 Pengujian Integrasi 29
2.13.3 Pengujian Validasi 30
2.13.4 Pengujian *Usability* 31
2.13.5 Pengujian Kompatibilitas 33
BAB 3 METODOLOGI 34
3.1 Tipe Dan Metode Penelitian 34
3.2 Model Proses Perangkat Lunak 34
3.3 Algoritma Yang Digunakan Pada Penelitian 36
3.3.1 *Ant Colony Optimization* 36
3.3.2 Algoritma *Brute Force* 36
3.4 Subjek Penelitian 36
3.5 Lokasi Penelitian 36
3.6 Perangkat Penelitian 37
3.7 Metode Pengumpulan Data 37
3.8 Strategi Pengujian dan Analisis Hasil Pengujian 38
BAB 4 REKAYASA PERSYARATAN 41

4.1 Gambaran Umum Sistem Informasi 41
4.2 Identifikasi Aktor 41
4.3 Analisis Data 42
4.4 Daftar Spesifikasi Persyaratan 42
4.5 Diagram *Use Case* 43
BAB 5 PERANCANGAN DAN IMPLEMENTASI 51
5.1 Perancangan Aplikasi 51
5.1.1 Perancangan Arsitektural 51
5.1.2 Perancangan *Activity Diagram* 52
5.1.3 Perancangan *Sequence Diagram* 55
5.1.4 Perancangan *Class Diagram* 56
5.1.5 Perancangan Algoritma 58
5.1.6 Perancangan Antarmuka 65
5.2 Implementasi 68
5.2.1 Spesifikasi Perangkat Keras 69
5.2.2 Spesifikasi Perangkat Lunak 69
5.2.3 Implementasi Algoritma 70
5.2.4 Implementasi Antarmuka 77
BAB 6 PENGUJIAN DAN ANALISIS 80
6.1 Pengujian 80
6.1.1 Pengujian Unit 81
6.1.2 Pengujian Integrasi 83
6.1.3 Pengujian Validasi Dengan Metode *Blackbox* 84
6.1.4 Pengujian *Usability* 86
6.1.3 Pengujian Kompatibilitas 88
6.2 Analisis 93
BAB 7 PENUTUP 95
7.1 Kesimpulan 95
7.2 Saran 96
Daftar Pustaka 97
LAMPIRAN 100

Daftar Gambar

Gambar 2.1 Jenis-Jenis *Graf* 7
Gambar 2.2 Jenis-Jenis *Graf Hamilton* 8
Gambar 2.3 Contoh *Graf* Lengkap Dengan 4 Simpul 9
Gambar 2.4 Perjalanan Semut Dari Sarang Menuju Sumber Makanan 10
Gambar 2.5 Metode *ACO K2-Opting* 14
Gambar 2.6 Pengaruh nilai α terhadap performa Algoritma ACO 14
Gambar 2.7 Pengaruh nilai β terhadap performa Algoritma ACO 15
Gambar 2.8 Pengaruh nilai ρ terhadap performa Algoritma ACO 15
Gambar 2.9 Perbandingan hasil ACO dengan nilai dari q0 16
Gambar 2.10 Perbandingan hasil ACO dengan nilai dari jumlah semut 17
Gambar 2.11 Arsitektur Android 23
Gambar 2.12 Contoh *Use Case Diagram* 26
Gambar 2.13 Contoh *Class Diagram* 27
Gambar 2.14 Contoh *Sequence Diagram* 27
Gambar 2.15 Pengujian Unit 29
Gambar 2.16 Pengujian Integrasi *Top-down* 29
Gambar 2.17 Pengujian Integrasi *Bottom-up* 30
Gambar 2.18 Perbandingan Hasil Skor SUS dengan Skor Akseptabilitas 33
Gambar 3.1 Model Proses Perangkat Lunak 35
Gambar 4.1 Diagram *Use Case* Sistem 43
Gambar 5.1 Perancangan Arsitektural Sistem 52
Gambar 5.2 *Activity Diagram* Mencari Rute Walisongo 53
Gambar 5.3 *Activity Diagram* melihat rute perjalanan 54
Gambar 5.4 *Activity Diagram* Melihat Informasi Walisongo 54
Gambar 5.5 *Sequence Diagram* Mencari Rute Walisongo 55
Gambar 5.6 *Sequence Diagram* Melihat Rute Walisongo 55
Gambar 5.7 *Sequence Diagram* Melihat Informasi Tentang Walisongo 56
Gambar 5.8 Class Diagram 57
Gambar 5.9 Antamuka Menu Utama 66
Gambar 5.10 Antarmuka Pilihan Menu Tujuan 67

Gambar 5.11 Tampilan Peta 67
Gambar 5.12 Tampilan Load Rute 68
Gambar 5.13 Tampilan Informasi Walisongo 68
Gambar 5.14 Parameter Algoritma *Ant Colony Optimization* 71
Gambar 5.15 Perhitungan probabilitas 72
Gambar 5.16 Mengambil Bilangan Random 72
Gambar 5.17 Tahap K2-Opting 74
Gambar 5.18 Perhitungan update lokal dan penguapan Feromon 74
Gambar 5.19 Perhitungan Update Feromon Global 74
Gambar 5.20 Parameter Algoritma *Brute Force* 75
Gambar 5.21 Rute terbaik 76
Gambar 5.22 mencoba kemungkinan semua rute 76
Gambar 5.23 Implementasi Antarmuka Menu 77
Gambar 5.24 Antarmuka Menu Penelusuran 78
Gambar 5.25 Implementasi Antarmuka Menu Load Rute 78
Gambar 5.26 Implementasi Antarmuka Informasi Walisongo 79
Gambar 6.1 *Flow Grap Method* doRute 81
Gambar 6.2 *Flow Graph Method* menampilkanRute 83

Daftar Tabel

Tabel 2.1 Skala SUS 32
Tabel 4.1 Identifikasi Aktor 42
Tabel 4.2 Spesifikasi Kebutuhan Fungsional 42
Tabel 4.3 Daftar kebutuhan non-fungsional 43
Tabel 4.4 Memilih Makam Walisongo 44
Tabel 4.5 Menentukan Lokasi Awal 45
Tabel 4.6 Melihat Tempat Walisongo 46
Tabel 4.7 Melakukan Penelusuran Tercepat 47
Tabel 4.8 Melihat Hasil Penelusuran 48
Tabel 4.9 Melihat Informasi Tentang Walisongo 49
Tabel 5.1 Koordinat awal dan Makam Walisongo 59
Tabel 5.2 Nilai Latitude Radian dan Longitude Radian 59
Tabel 5.3 Jarak Antar Kota 60
Tabel 5.4 Visibilitas Antar Kota 60
Tabel 5.5 Feromon Awal Antar Titik 60
Tabel 5.6 Hasil Perhitungan Σ 61
Tabel 5.7 Perhitungan Probabilitas Semut Menuju Kota Selanjutnya 62
Tabel 5.8 Pembaruan Feromon Lokal 63
Tabel 5.9 Hasil Pencarian Rute 63
Tabel 5.10 Pembaruan Feromon Global 64
Tabel 5.11 Daftar Titik Setiap Rute Walisongo 65
Tabel 5.12 Hasil Pencarian Rute Walisongo 65
Tabel 5.13 Spesifikasi Perangkat Keras 69
Tabel 5.14 Spesifikasi Perangkat Lunak 70
Tabel 6.1 Pengujian Unit *Method* doRute 81
Tabel 6.2 Kasus Uji Untuk Pengujian Unit *method* doRute 82
Tabel 6.3 Pengujian Unit *Method* menampilkanRute 83
Tabel 6.4 Kasus Uji Untuk Pengujian Integrasi *Method* menampilkanRute 84
Tabel 6.5 Hasil pengujian validasi sistem informasi Walisongo 85
Tabel 6.6 Pertanyaan Pengujian *Usability* Kemudahan Penggunaan 86

Tabel 6.7 Pertanyaan Pengujian *Usability* Hasil Pencarian Rute 87
Tabel 6.8 Pengujian Smartphone 91
Tabel 6.9 Pengujian Tablet 92

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Walisongo dikenal sebagai penyebar agama Islam pada abad 14 di pulau Jawa. Para Walisongo tinggal di wilayah penting pantai utara Pulau Jawa, yaitu Surabaya,Gresik,Lamongan,Tuban di Jawa Timur, Demak, Kudus, Muria di Jawa Tengah, dan Cirebon di Jawa Barat. Saat ini ziarah Walisongo menjadi tradisi di Indonesia. Pengurus Kompleks Makam Wisata Ziarah Syekh Maulana Malik Ibrahim, Salim menjelaskan, rata-rata pengunjung berkelompok mencapai 100 bus per hari. Jika satu bis isi 50 orang rata-rata di bulan Syaban ada 5.000 pengunjung yang datang. Itu belum termasuk yang datang dengan kendaraan pribadi atau angkutan umum, tuturnya. Pada tahun 2010 jumlah pengunjung mencapai 1.365.000 pengunjung lokal dan 400.000 pengunjung manca negara. Angka itu naik sekitar 15 persen dari tahun sebelumnya (Adi & Anwar, 2011).

Dengan banyaknya obyek yang dikunjungi, pemilihan rute sangat diperlukan sehingga perjalanan menjadi efisien dan cepat. Secara matematis kondisi seperti ini dapat direpresentasikan sebagai sebuah *graf*. *Graf* adalah pasangan himpunan *vertex* atau simpul dan *edges*/sisi, dimana setiap sisi berhubungan dengan satu atau dua buah simpul (Munir, 2010).

Masalah pemilihan rute dapat dianalogikan dengan *Travelling Salesman Problem* (TSP). TSP merupakan sebuah masalah yang biasa dihadapi oleh seorang *salesman* dalam rangka menempuh perjalan dari suatu kota ke n-kota lain tepat satu kali dan kembali ke kota awal keberangkatan. Algoritma *Ant Colony Optimization* merupakan salah satu algoritma yang dapat menyelesaikan masalah ini karena jumlah kota relatif sedikit. Jika dibandingkan dengan algoritma Genetika, Jika jumlah kota relatif banyak waktu pencarian algoritma genetika lebih cepat dibandingkan algoritma semut sedangkan untuk jumlah kota relatif kecil waktu pencarian algoritma semut lebih cepat (Siswoyo & Andrianto, 2009). Setelah rute didapat oleh algoritma Ant Colony Optimization maka selanjutnya tugas algoritma Brute Force untuk mencari rute yang paling tercepat karena algoritma Brute Force memiliki sifat langsung ke pusat permasalahan.

Oleh karena itu perlu dirancang aplikasi yang dapat menunjukkan rute wali yang akan dikunjungi terlebih dahulu dan dapat menentukan jalur wali-wali selanjutnya yang akan dikunjungi dengan berdasarkan lokasi awal pengguna secara efektif dan dapat membaca hasil penelusuran yang telah dicari. Selain itu aplikasi ini akan memberikan informasi tentang sejarah para Walisongo untuk memberi wawasan para pengguna.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, maka dapat dirumuskan permasalahannya yaitu :

1. Bagaimana mengimplementasikan *Location Based Service* pada perangkat bergerak untuk informasi Walisongo ?
2. Bagaimana mengimplementasikan algoritma *Ant Colony Optimization* dan algoritma *Brute Force* untuk mencari rute tercepat ?
3. Bagaimanakah hasil pengujian unit, integrasi, validasi dan usabilitas dan kompatibilitas pada aplikasi walisongo sudah sesuai dengan perancangan?

1.3 Batasan Masalah

Agar permasalahan yang dirumuskan lebih terfokus dan tidak terjadi pelebaran topik, maka penelitian tugas akhir ini dibatasi dalam hal :

1. Informasi yang ditampilkan adalah informasi mengenai rute perjalanan, nama tempat, dan sejarah Walisongo.
2. Perangkat lunak untuk sistem informasi Makam Walisongo menggunakan Algoritma *Ant Colony Optimization* dan Algoritma *Brute Force*.
3. Pengambilan data peta dari *Google Maps API*
4. Sistem informasi Makam Walisongo menggunakan perangkat bergerak berbasis Android minimal versi 2.3 (*Gingerbread*) dengan resolusi layar minimal 480x800 (*WVGA*).
5. Pengujian akan menggunakan pengujian unit, integrasi, validasi, *usability* dan kompatibilitas.

1.4 Tujuan

Tujuan dari pembuatan aplikasi sistem informasi Walisongo ini secara umum untuk merancang, menganalis, membangun, dan menguji sistem informasi yang berhubungan dengan rute Walisongo dengan menggunakan metode algoritma *Ant Colony Optimization*.

1.5 Manfaat

A. Bagi Pengguna Sistem

1. Aplikasi ini diharapkan dapat digunakan oleh masyarakat umum, para peziarah dan pengujung Walisongo untuk mempermudah dalam mendapatkan informasi secara lengkap, akurat dan praktis mengenai informasi para Walisongo

B. Bagi Penulis

1. Memperdalam ilmu pengetahuan yang berkaitan dengan pembuatan aplikasi pada perangkat bergerak.
2. Memperdalam ilmu pengetahuan mengenai pembuatan aplikasi yang menggunakan teknologi *Location Based Service*.
3. Memperdalam ilmu pengetahuan mengenai algoritma *Ant Colony Optimization*.

1.6 Sistematika Pembahasan

Sistematika penulisan laporan skripsi ini sebagai berikut :

- Bab I Pendahuluan

  Bab Pendahuluan berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

- Bab II Dasar Teori

  Bagian ini membahas tentang dasar teori secara luas mengenai LBS (*Location Based Service*) dan teori pendukungnya untuk membangun sebuah sistem informasi dengan LBS di perangkat bergerak berbasis Android.

- Bab III Metodologi Penelitian

  Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam perancangan, implementasi dan pengujian dari aplikasi rancang bangun aplikasi informasi walisongo berbasis *Location Based Service* Pada *Smartphone* Android.

- Bab IV Rekayasa Persyaratan

  Bab ini akan menjelaskan tentang gambaran umum sistem informasi, identifikasi aktor, analisis data, daftar spesifikasi persyaratan dan pemodelan diagram use case yang akan berguna untuk membuat perangkat lunak yang sesuai.

- Bab V Perancangan dan Implementasi

  Bab ini menjelaskan realisasi dari perancangan dan pengumpulan teori mengenai sistem informasi yang akan dibuat.

- Bab VI Pengujian

  Bab ini berisi pengujian pada perangkat bergerak untuk mengetahui kinerja dari aplikasi yang dibuat.

- Bab VII Penutup

  Bab ini berisi kesimpulan yang diambil berdasarkan analisa hal-hal penting, meliputi keunikan, kelebihan atau kekurangan serta saran-saran untuk penyempurnaan sistem yang dibuat.

BAB 2 LANDASAN TEORI

Pada bab dua terdiri dari kajian pustaka dan dasar teori. Kajian pustaka berisi tentang penelitian yang telah ada dan yang diusulkan. Dasar teori membahas teori yang diperlukan untuk menyusun skripsi ini. Kajian pustaka pada penelitian ini adalah mengembangkan skripsi ini dari skripsi sebelumnya yang berjudul "Pencarian Rute Angkutan Umum Menggunakan Algoritma Ant Colony Optimization" oleh Candra Irwansyah. Pada skripsi ini, dasar teori yang diperlukan berdasarkan latar belakang dan rumusan masalah.

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini adalah mengembangkan skripsi ini dengan skripsi sebelumnya yang berjudul "Pencarian Rute Angkutan Umum Menggunakan Algoritma Ant Colony Optimization" oleh Candra Irwansyah. Aplikasi Rute Angkutan Kota Malang memanfaatkan algoritma Ant Colony Optimization dan algoritma Brute Force lalu diterapkan untuk menentukan angkutan umum apa yang akan digunakan. Data yang digunakan dalam skripsi ini adalah data rute angkutan umum yang ada di kota Malang. Data angkutan umum kemudian dibuat menjadi sebuah *vertex* jarak yang nantinya akan di proses oleh algoritma *Ant Colony Optimization* dan algoritma *Brute Force*. Hasil pengujian menunjukkan bahwa waktu pemrosesan untuk pencarian jarak dekat yang memiliki jumlah kombinasi rute angkutan umum lebih banyak memiliki waktu pemrosesan yang lebih lama dibandingkan pencarian jarak jauh yang memiliki jumlah kombinasi rute angkutan umum lebih sedikit (Irwansyah, 2014).

Skripsi kedua adalah "Algoritma *Ant Colony Optimization* (ACO) Untuk Menyelesaikan *Traveling Salesman Problem* (TSP)" oleh Agus Leksono. *Traveling Salesman Problem* (TSP) merupakan sebuah persoalan penting dalam sistem distribusi. Masalah *traveling salesman* secara umum digambarkan sebagai suatu kasus dimana seseorang harus mengunjungi sejumlah kota dari suatu pusat fasilitas dan kembali lagi ke tempat pemberangkaan semula, dengan asumsi jarak diketahui. Untuk menyelasaikan masalah tersebut biasa digunakan algoritma heuristik. Algoritma *Ant Colony Optimization* (ACO) merupakan salah satu metedo metaheuristik yang menerapkan semut sebagai agen dengan update *pheromone*-nya untuk dapat melakukan proses pencarian solusi yang efektif dan efisien. Algoritma ACO yang dibandingkan pada skripsi ini sebanyak lima yaitu *Ant System* (AS), *Elitist Ant System* (EAS), *Rank-based Ant System* (AS_{Rank}), *Max-min Ant System* (MMAS), dan *Ant Colony System* (ACS). Simulasi dilakukan dengan mencari solusi mendekati optimal dari beberapa kasus TSP dengan jumlah titik n = 20 sampai n = 115. Hasil mendekati optimal diperoleh dengan melakukan beberapa kali percobaan untuk setiap kasuk dan setiap algoritma dibandingkan. Hasil perbandingan kelima Algoritma ACO tersebut dapat terlihat bahwa untuk jumlah titik sampai n =40 solusi yang dihasilkan semua algoritma sama. Untuk kasus dengan jumlah titik yang lebih banyak, algoritma ACS mempunyai solusi yang

terbaik dan Algoritma AS yang terjelek dari kelima algoritma tersebut (Leksono, 2009).

Banyak aplikasi pencarian rute yang memanfaatkan algoritma *Ant Colony Optimization* dan algoritma *Brute Force* namun hingga saat ini belum ada aplikasi pencarian rute yang menyelesaikan permasalahan para peziarah untuk mengetahui rute tercepat ke makam para Walisongo.

2.2 Pengertian Informasi

Informasi adalah rangkaian data yang mempunyai sifat sementara, tergantung dengan waktu, mampu memberikan kejutan atau *surprise* pada yang menerimanya. Intensitas dan lamanya kejutan dari informasi, disebut nilai informasi yang tidak mempunyai nilai, biasanya karena rangkaian data yang tidak lengkap atau kadaluarsa. Informasi dapat juga dikatakan sebagai data yang telah diproses, yang mempunyai nilai tindakan atau keputusan. (Witarto, 2004)

2.3 LBS (Location Based Service)

Location Based Service (LBS) adalah salah satu bagian dari penerapan mobile GIS yang lebih cenderung memberikan fungsi terapan sehari-hari. Seperti menampilkan direktori kota, navigasi kendaraan, pencarian alamat serta jejaring sosial dari pada fungsi pada teknologi GIS populer untuk Field Based GIS (Riyanto, 2010). Dua unsur utama LBS adalah :

- *Location Manager* (API Maps) : Menyediakan alat berupa *Application Programming Interface* (API) Maps yang memeliki fungsi untuk menampilkan, memanipulasi peta beserta fungsi lainnya seperti tampilan satelit, jalan, maupun keduanya.
- *Location Providers* (API *Location*) : Menyediakan fungsi pencarian lokasi yang digunakan oleh perangkat. API *Location* berhubungan erat dengan data dari GPS dan lokasi *real time*. Dengan *Location Manager*, dapat menentukan lokasi pengguna saat ini dan rute menuju tempat tertentu.

2.4 Graf

Graf merupakan struktur matematika diskrit yang terdiri dari sekumpulan simpul (*node*) dan sisi (*edges*) yang secara matematis dituliskan sebagai *G(V,E)* [KER-99]. Secara geometri, graf juga dapat digambarkan sebagai sekumpulan *node* (simpul) pada bidang dwimatra yang dihubungkan dengan sekumpulan edges /(sisi) (Munir, 2010).

2.4.1 Jenis-Jenis Graf

Berdasarkan ada dan tidaknya sisi gelang (*loop*) atau sisi ganda (*multiple / parallel edges*) pada suatu graf dapat dikelompokan menjadi dua yaitu (Munir, 2010) :

1. Graf sederhana (*Simple Graph*)

Graf Sederhana merupakan graf yang tidak memiliki sisi gelang maupun sisi ganda. Sisi gelang merupakan sisi yang berawal dan berakhir pada simpul yang sama, sedangkan sisi ganda merupakan sisi-sisi yang menghubungkan dua buah simpul yang sama.

2. Graf tak sederhana (*unsimple graph*)

Graf tak sederhana merupakan graf yang memiliki gelang atau sisi ganda. Ada dua macam graf tak sederhana, yaitu graf semu (*pseudograph*) dan graf ganda (*multigraph*). Graf semu lebih umum daripada graf ganda. Karena sisi pada graf semu dapat terhubung ke dirinya sendiri (membentuk gelang).

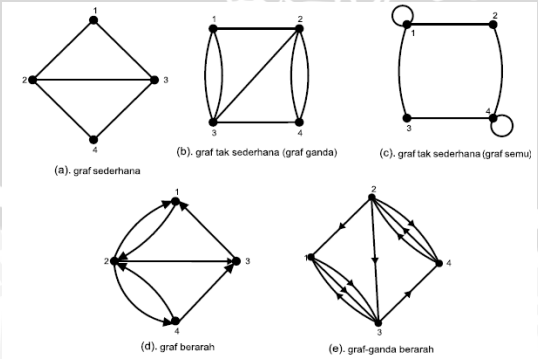
jika berdasarkan orientasi arah pada sisi, maka graf dapat dibedakan menjadi dua, yaitu (Munir, 2010) :

1. Graf tak berarah (*Undirected graf*)

Graf tak berarah adalah graf yang tidak mempunyai orientasi arah, sehingga urutan pasangan simpul yang dihubungkan oleh sisi tidak diperhatikan, maka (*Vj,Vk*) = (*Vk,Vj*) merupakan sisi yang sama.

2. Graf Berarah (*Directed graf*)

Graf berarah yaitu graf yang setiap sisinya diberikan orientasi arah. Pada graf berarah, sisi (*Vj,Vk*) dan (*Vk,Vj*) menyatakan dua sisi yang berbeda. Graf berarah seing dipakai untuk menggambarkan aliran proses, peta lalu lintas, dan lain-lain. Pada graf berarah, gelang diperbolehkan tetapi sisi ganda tidak. Gambar graf ditunjukkan dalam gambar 2.1



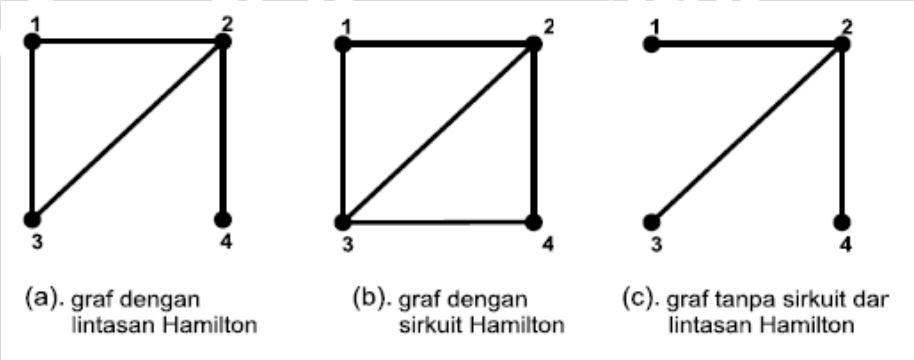
Gambar 2.1 Jenis-Jenis *Graf*

Sumber : Munir (2010)

2.4.2 Lintasan dan Sirkuit Hamilton

Lintasan Hamilton adalah sebuah lintasan yang melalui tiap simpul didalam graf sebanyak satu kali. Bila lintasan itu kembali ke simpul asal sehingga membentuk lintasan tertutup yang biasa disebut sebagai Sirkuit Hamilton. Jadi Sirkuit Hamilton adalah sirkuit yang melalui tiap simpul didalam graf sebanyak satu kali, kecuali simpul awal (sekaligus simpul akhir) yang dilalui dua kali (Munir, 2010).

Graf yang memiliki Sirkuit Hamilton dinamakan sebagai Graf Hamilton. Sedangkan graf yang hanya memiliki Lintasan Hamilton disebut Graf semi-Hamilton. Setiap graf lengkap juga disebut Graf Hamilton (Munir, 2010). Gambar graf Hamilton ditunjukkan pada gambar 2.2



Gambar 2.2 Jenis-Jenis *Graf Hamilton*

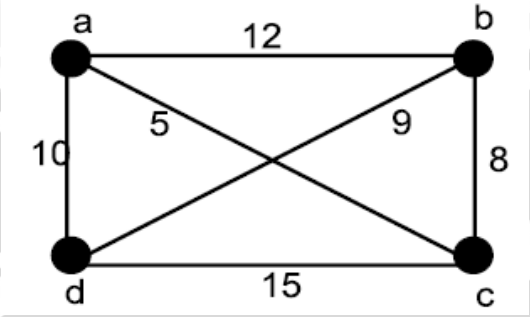
Sumber : Munir (2010)

2.5 *Travelling Salesman Problem* (TSP)

TSP adalah sebuah optimasi permasalahan kombinatorial yang sangat terkenal dalam teori graf. TSP dikategorikan sebagai permasalahan yang sulit ditinjau dari sudut komputasinya (Munir, 2010). TSP juga termasuk sebuah permasalahan *NP-Complete* yang klasik karena telah dipelajaro selama beberapa tahun terakhir. TSP juga sebuah masalah dalam mencari rute terpendek yang harus ditempuh oleh seseorang yang berangkat dari kota asal untuk mengunjungi setiap kota tepat satu kali lalu kembali lagi ke kota asal keberangkatannya.

Dalam teori graf, TSP dinyatakan sebagai graf $G_{TSP}= (N,A)$ dimana *N* menyatakan *Node* atau himpunan simpul kota, sedangkan *A* himpunan sisi yang menghubungkan simpul-simpul kota pada graf. Bobot pada sisi menyatakan jarak antar dua buah kota. Dengan kata lain, TSP tidak lain adalah permasalahan untuk menemukan Sirkuit Hamilton yang memiliki bobot minimum yang memiliki bobot minimum dan terhubung pada sebuah graf (Munir, 2010).

TSP pada sembarang graf lengkap, sangat mudah untuk menghitung jumlah Sirkuit Hamilton yang ada. Dari simpul pertama ada *n*-1 pilihan, dari simpul kedua ada *n*-2 pilihan, dari simpul ketiga ada *n*-3 pilihan, dan seterusnya. Sehingga untuk *n* buah kota, ada (*n*-1)! Pilihan yang mungkin. Namun karena Sirkuit Hamilton terhitung 2 kali maka pilihan yang ada harus dibagi dua menjadi $\frac{(n-1)!}{2}$ buah Sirkuit Hamilton. Sebagai contoh dapat dilihat pada gambar 2.3



Gambar 2.3 Contoh Graf Lengkap Dengan 4 Simpul

Sumber : Munir (2010)

Untuk TSP pada graf lengkap dengan *n* = 4 seperti pada gambar 2.4, hasilnya terdapat $\frac{(4-1)!}{2} = 3$ Sirkuit Hamilton, yaitu :

1. S1 = (*a,b,c,d,a*) atau (*a,d,c,b,a*) dengan panjang rute = 12 + 8 + 15 + 10 = 45
2. S2 = (*a,c,d,b,a*) atau (*a,b,d,c,a*) dengan panjang rute = 5 + 15 + 9 + 12 = 41
3. S3 = (*a,c,b,d,a*) atau (*a,d,b,c,a*) dengan panjang rute = 5 + 8 + 9 + 10 = 32

Hasilnya, Sirkuit Hamilton terpendek adalah S3 = (*a,c,b,d,a*) atau (*a,d,b,c,a*) dengan panjang rute sebesar 32.

TSP juga dapat dipecahkan dengan melakukan *exhaustive search* pada $\frac{(n-1)!}{2}$ Sirkuit Hamilton dengan menghitung panjang rute masing-masing lalu menentukan rute terpendek. Namun untuk *n* yang besar, maka jumlah Sirkuit Hamilton yang harus diperiksa satu persatu akan banyak. Sebagai contoh, untuk *n* = 20 akan ada sekitar 6 x 10^{16} penyelesaian. Hasilnya, waktu komputasi yang diperlukan juga akan meningkat (kompleksitasnya *O(n!)*). Maka dari itu TSP termasuk kategot permasalahan yang *NP-Complete* (Munir, 2010).

2.6 Algoritma *Ant Colony Optimization*

Ant Colony Optimization diadopsi dari perilaku semut. Semut mampu beradaptasi ke dalam lingkungannya untuk mencari makanan dan kemudian kembali ke sarangnya dengan meninggalkan feromon pada rute-rute yang telah mereka lewati. (Dorigo & Stutzle, 2002)

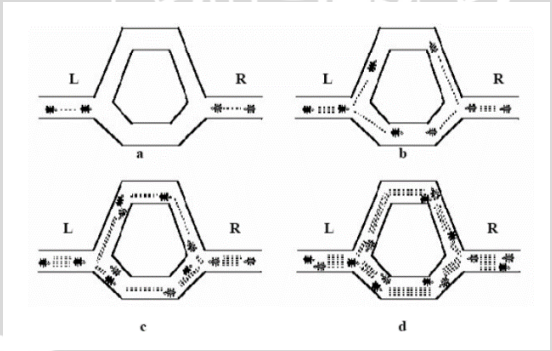
Feromon adalah zat kimia yang berasal dari kelenjar endokrin yang digunakan oleh makhluk hidup untuk mengenali sesama jenisnya, individu lain, kelompok, dan juga berguna untuk membantu proses reproduksi. Feromon berbeda dengan hormon karena feromon menyebar ke luar tubuh dan hanya dapat mempengaruhi dan dikenali oleh sesama spesies. (Dorigo & Stutzle, 2002)

Proses peninggalan feromon ini juga dikenal dengan *stigmery*. *Stigmery* sendiri adalah sebuah proses mengubah suatu lingkungan yang tidak hanya digunakan untuk mengingat jalan pulang ke sarang, tetapi juga memungkinkan para semut dapat berkomunikasi dengan koloninya. (Dorigo & Stutzle, 2002)

Seiring dengan waktu, jejak feromon akan menguap dan akan mengurangi kekuatan daya tariknya. Lebih cepat setiap semut pulang pergi melalui rute tersebut, maka feromon yang menguap lebih sedikit. Berbeda jika semut lebih lama pulang pergi melalui rute tersebut, maka feromon yang menguap lebih banyak. (Dorigo & Stutzle, 2002)

2.6.1 Cara Kerja Semut Menemukan Rute Terpendek Dalam ACO

Secara alamiah semut dapat menemukan rute terpendek dalam perjalanan dari sarang menuju sumber makanan. Koloni semut dapat menemukan rute terpendek antara sarang dan sumber makanan berdasarkan jejak kaki pada lintasan yang sudah dilalui. Semakin banyak semut yang melalui suatu lintasan, maka semakin jelas pula bekas jejak kakinya. Hal ini yang menyebabkan lintasan yang dilalui semut dalam jumlah sedikit maka semakin lama akan semakin berkurang jumlah semut yang melewatinya. Sebaliknya lintasan yang dilalui semut dalam jumlah banyak maka semakin lama akan semakin bertambah jumlah semutnya, bahkan semua semut akan melalui lintasan tersebut. (Dorigo & Gambardella, 1997)



Gambar 2.4 Perjalanan Semut Dari Sarang Menuju Sumber Makanan

Sumber : Dorigo & Gambardella (1997)

Gambar 2.4.a menunjukkan ada dua kelompok semut yang akan melakukan perjalanan. Satu kelompok bernama L yang berangkat dari sarang semut dan yang berangkat dari arah kanan bernama R yang berasal dari sumber

makanan. Kedua kelompok ini sedang berada dalam posisi pengambilan keputusan jalan sebelah mana yang akan dilalui. Kelompok semut L membagi dua kelompok lagi, sebagian melalui jalan atas dan sebagian akan melalui jalan bawah. Hal ini juga berlaku pada kelompok semut R. Gambar 2.4.b dan gambar 2.4.c menunjukkan bahwa kelompok semut berjalan pada kecepatan yang sama dan meninggalkan feromon (jejak kaki semut) di jalan yang dilalui. Feromon yang ditinggalkan oleh semut-semut melalui jalan atas berjumlah sedikit dibandingkan dengan jalan yang berada di bawah. Hal ini dikarenakan jarak jalan atas ditempuh lebih panjang daripada jalan bawah. Feromon yang berada dijalan bawah pengupannya lebih lama, karena semut yang melalui jalan bawah lebih banyak daripada semut yang melalui jalan atas. Gambar 2.4.d menunjukkan bahwa pada akhirnya semut-semut yang lain memilih untuk melewati jalan bawah karena feromon yang ditinggalkan masih banyak. Sedangkan feromon yang berada pada jalan atas semakin banyak menguap sehingga semut-semut tidak memilih jalan atas. Jadi semakin banyak semut yang melalui jalan bawah maka semakin banyak pula yang mengikutinya. (Dorigo & Gambardella, 1997)

Demikian juga dengan jalan atas, semakin sedikit yang melewati jalan atas, maka feromonnya semakin berkurang bahkan menghilang. Dari sinilah kemudian terpilih rute terpendek antara sarang dan sumber makanan. (Dorigo & Gambardella, 1997)

Inisialisasi parameter – parameter algoritma

1. Intensitas jejak semut antar kota dan perubahannya (τij)
2. Banyak kota (n) termasuk koordinat (x,y) atau jarak antar kota (dij) serta kota awal dan kota tujuan
3. Tetapan siklus-semut (Q)
4. Tetapan pengendali intensitas jejak semut (α), nilai $\alpha \geq 0$
5. Tetapan pengendali visibilitas (β), nilai $\beta \geq 0$.
6. Visibilitas antar kota = 1/dij (ηij)
7. Banyak semut (m)
8. Tetapan penguapan jejak semut (ρ) , nilai ρ harus > 0 dan < 1 untuk mencegah jejak feromon menjadi tak terhingga
9. Jumlah siklus maksimum (Ncmax) bersifat tetap selama algoritma dijalankan, sedangkan τij akan selalu diperbarui nilainya pada setiap siklus algoritma yang dimulai dari siklus awal (NC=1) sampai tercapai jumlah siklus maksimum (NC = Ncmax) atau sampai terjadi konvergensi.

Aturan transisi digunakan oleh sistem semut, disebut juga dengan *random-proportional rule* diberikan oleh persamaan (2.1), yang memberikan probabilitas semut *k* di kota *r* memilih untuk pindah ke kota *s.* (Dorigo & Stutzle, 2004)

$$p_k(r,s) = \left\{ \frac{[\tau(r,s)]^{\alpha}.[\eta(r,s)]^{\beta}}{\sum_{u \epsilon jk(r)}[\tau(r,u)]^{\alpha}.[\eta(r,u)]^{\beta}} \right\} \text{ Jika s } \epsilon \text{ } J_k\,(r) \quad (2.1)$$

Dimana :

$P_k\,(r,s)$: Probabilitas semut k memilih untuk berpindah dari kota r ke kota s

$\tau\,(r,s)$: Jumlah feromon pada sisi dari simpul r ke simpul s

$\eta\,(r,s)$: (panjang sisi dari simpul r ke simpul u)$^{-1}$

$\tau\,(r,u)$: jumlah feromon pada sisi dari simpul r ke simpul u.

$\eta(r,u)$: (Panjang sisi dari simpul r ke simpul u)$^{-1}$

J_k : Himpunan yang berisi simpul – simpul yang telah dikunjungi oleh semut

s : simpul yang berada dalam J_k

Dalam ACS (*Ant Colony System*) aturan transisi status adalah sebagai berikut. Semut di posisikan pada node *r* lalu memilih kota *s* untuk berpindah dengan menerapkan aturan yang diberikan oleh persamaan (2.2). (Dorigo & Stutzle, 2004)

$$s = \{\max\{[\tau(r,u)] \cdot [n(r,u)]^{\beta}\}\} \qquad jika\ q \leq q_0 \quad (eksploitasi) \quad (2.2)$$

$$S = P_k\,(r,s) \qquad jika\ q > q_0 \quad (eksplorasi)$$

Dimana :

$\tau(r,u)$: Jumlah feromon pada sisi dari simpul r ke simpul s

$n(r,u)$: (panjang sisi dari simpul r ke simpul s) $^{-1}$

β : parameter perbandingan jumlah feromon relatif terhadap jarak (parameter sudah ditentukan sebelumnya)

q : bilangan random

q0 : parameter pembanding nilai random terhadap titik yang belum ditemuinya (probabilitas semut melakukan eksplorasi)

S : titik berikutnya yang dipilih berdasarkan persamaan (2.1)

Jika $q \leq q_0$ maka semut akan memanfaatkan fungsi heuristik mengenai jarak antar titik dengan titik-titik lainnya yang nantinya akan disimpan ke dalam bentuk feromon. Hal ini menghasil *edge* terbaik. Jika $q > q_0$ maka mengunakan persamaan p_k (r,s) (persamaan 2.1). Aturan transisi pada sistem ini memberikan suatu cara langsung untuk menyeimbangkan antara penjelajahan *edge-edge* yang baru dengan eksploitasi. (Dorigo & Stutzle, 2004)

Dalam ACS hanya semut terbaik secara keseluruhan (yaitu pada saat perjalan semut yang terpendek dari awal berjalan) yang diperbolehkan untuk meninggal feromon. Pilihan ini, bersama-sama dengan penggunaan aturan *pseudo random*

proportional yang berguna untuk membuat pencarian agar lebih terarah. Semut akan mencari rute perjalanan terbaik ditemukan hingga akhir iterasi algoritma. Pembaruan feromon global dilakukan setelah semua semut selesai melakukan rute perjalanan mereka. Tingkat feromon diperbarui dengan menggunakan aturan memperbarui persamaan global (2.3). (Dorigo & Stutzle, 2004)

$$\tau(r,s) \leftarrow (1-\alpha) \cdot \tau(r,s) + \alpha\Delta\tau(r,s) \qquad (2.3)$$

Dimana :

$\tau(r,s)$: nilai feromon akhir setelah mengalami perubahan

α : tetapan pengendali feromon

$\Delta\tau$: perubahan intensitas feromon

Pembaruan feromon global ini adalah untuk menyediakan besaran feromon untuk mengunjungi rute terpendek. Persamaan 2.3 menyatakan bahwa hanya sebuah *edge* rute perjalanan terbaik secara keseluruhan akan mendapat penguatan nilai feromon. (Dorigo & Stutzle, 2004)

Ketika membuat sebuah solusi TSP, semut akan mengunjungi *edge* dan mengubah tingkat feromonnya dengan menerapkan aturan memperbarui feromon lokal. Persamaan (2.4) (Dorigo & Stutzle, 2004)

$$\tau(r,s) \leftarrow (1-\rho) \cdot \tau(r,s) + \rho \cdot \Delta\tau(r,s) \qquad (2.4)$$

Dimana :

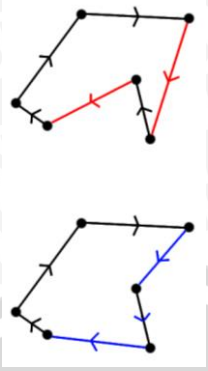
$\tau(r,s)$: jumlah feromon pada sisi dari simpul r ke simpul s

ρ : tetapan penguapan feromon

$\Delta\tau$: perubahan intensitas feromon (τ_0)

Aturan memperbarui feromon lokal adalah untuk mengacak rute, sehingga kota-kota di awal pada rute seekor semut dapat dieksplorasi selanjutnya oleh rute semut lain. Sehingga efek dari pembaruan feromon lokal untuk membuat *edge* berubah secara dinamis. Setiap semut menggunakan *edge* ini tidak akan dilewati oleh semut lain dan mulai kehilangan intensitas feromon sehingga semut akan mencari rute tercepat dan menghasilkan intensitas feromon akan meningkat. (Dorigo & Stutzle, 2004)

Namun ada cara lain dalam menentukan rute menurut *gebweb optimap*, yaitu *ACO K2-Opting*. *ACO K2-opting* digunakan ketika semut telah menyelesaikan perjalanannya. Nantinya akan dipilih dua sisi dari rute dan dilakukan pengecekan apakah sisi-sisi tersebut dapat ditukarkan sehingga menjadi rute yang lebih baik. Seperti pada gambar 2.5 (Gebweb, 2007)



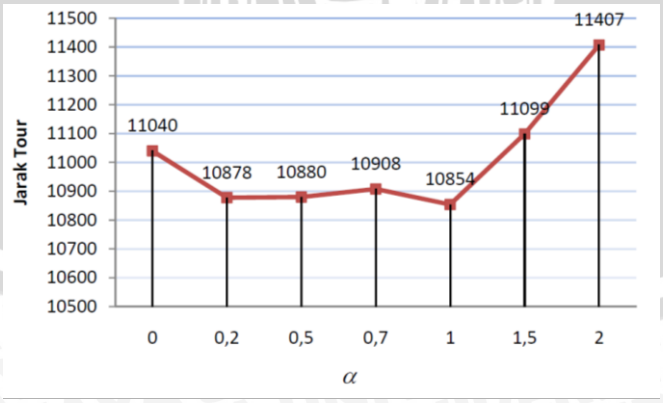
Gambar 2.5 Metode *ACO K2-Opting*

Sumber : Gebweb (2007)

Prosedur ini diulang hingga tidak ada dua sisi yang dapat ditukarkan untuk menghasilkan rute tercepat. Cara ini dapat meningkatkan banyak metode heuristik dari TSP (Gebweb, 2007)

2.6.2 Penetapan Parameter Algoritma *Ant Colony Optimization*

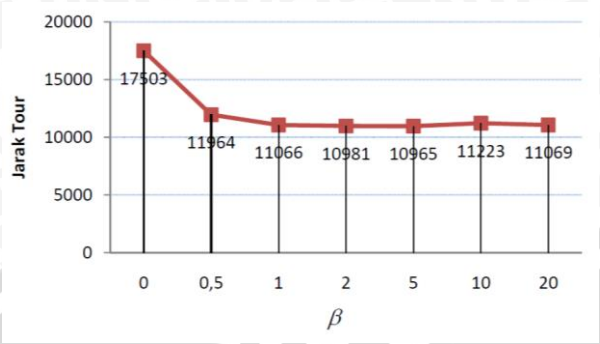
Eksperimen yang dilakukan oleh Dorigo, M., Manezzo, V., dan Coloni, A, (1991) yang dilakukan dengan algoritma *Ant Colony* untuk menyelesaikan TSP, parameter - paramater α, β, dan ρ mempunyai pengaruh terhadap hasil yang diperoleh dari algoritma *Ant Colony Optimization*. Sehingga untuk mengetahui pengaruh dari parameter – parameter tersebut, Berikut hasil simulasi dengan ACOTSP, versi 1.0 yang dikembangkan oleh dorigo dan stutzle. Permasalahan yang diujikan yaitu att48 dengan α, β, dan ρ masing – masing dengan nilai : $\alpha \in \{0; 0{,}2; 0{,}5; 0{,}7; 1; 1{,}5; 2\}$, $\beta \in \{0; 0{,}5; 1; 2; 5; 10; 20\}$ dan $\rho \in \{0{,}1; 0{,}3; 0{,}5; 0{,}7; 0{,}9;1\}$. Hasil eksperimen yang diperoleh ditunjukkan pada gambar 2.6 untuk nilai α, gambar 2.7 untuk nilai β, dan gambar 2.8 untuk nilai ρ. (Leksono,2009)



Gambar 2.6 Pengaruh nilai *α* terhadap performa Algoritma *Ant Colony Optimization*

Sumber : Leksono (2009)

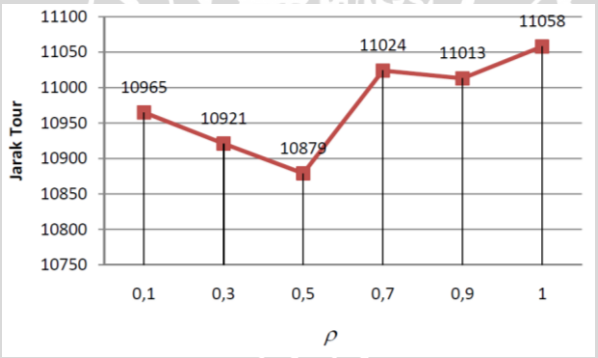
Dari gambar 2.6 menunjukkan dengan nilai α = 1 diperoleh hasil rute terbaik sebesar 10854 dan hasil rute paling buruk ditunjukkan dengan nilai α = 2 dengan panjang rute sebesar 11407.



Gambar 2.7 Pengaruh nilai *β* terhadap performa Algoritma *Ant Colony Optimization*

Sumber : Leksono (2009)

Dari gambar 2.7 menunjukkan dengan nilai β = 5 diperoleh hasil yang terbaik dengan panjang rute 10965 dan hasil yang paling buruk diperoleh dengan nilai β = 0 dengan panjang rute 17503. Meskipun hasil terbaik adalah nilai β = 5, namun hasil dari nilai β = 2 sangat mendekati hasil terbaik dengan panjang rute 10981. Jadi dengan hasil terbaik pada algoritma *ant colony optimization* bisa diperoleh dengan nilai β = 2 sampai dengan β = 5.



Gambar 2.8 Pengaruh nilai *ρ* terhadap performa Algoritma *Ant Colony Optimization*

Sumber : Leksono (2009)

Dari hasil diatas menunjukkan dengan nilai ρ = 0.5 diperoleh hasil yang terbaik dengan panjang rute = 10879. Sedangkan yang terburuk diperoleh nilai ρ = 1 dengan panjang rute terbaiknya 11058. (Leksono,2009)

Dapat disimpulkan dari ketiga parameter diatas maka menghasilkan kombinasi yang paling bLeksono untuk memperoleh hasil yang mendekati terbaik. Masing – masing nilai dari ketiga parameter tersebut adalah sebagai berikut. : α =

1, β = 2 sampai β = 5, dan ρ = 0.5. Pada beberapa kasus yang ada, kombinasi ini tidak selalu menghasilkan nilai yang mendekati terbaik. Hal ini tergantung dari besarnya masalah yang diselesaikan. (Leksono,2009)

Namun dalam setiap percobaan yang dilakukan dengan algoritma *Ant Colony Optimization* untuk menyelesaikan permasalah TSP, parameter yang digunakan untuk mendapatkan solusi yang lebih mendekati optimal masing – masing mempunyai nilai sebagai berikut : β = 2 sampai 5, *pseudo random propotional rule* nilai q_0 = 0.9, ρ = 0.1, dan t_0 yang dihitung dengan menggunakan persamaan 2.5. (Leksono, 2009)

$$\tau_0 = \frac{1}{(cL^{nn})} \qquad (2.5)$$

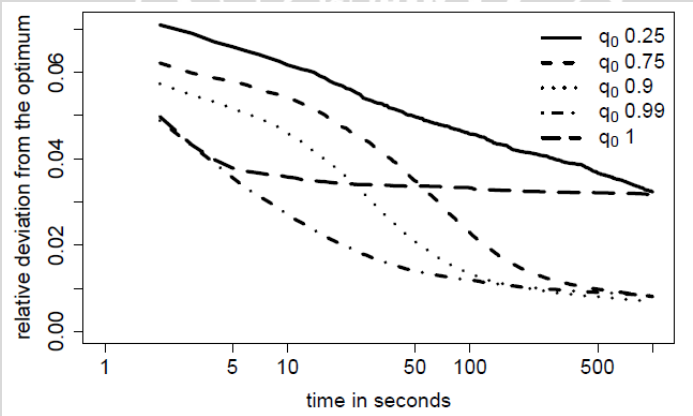
Dimana :

L^{nn} = Panjang rute terbaik yang diperoleh dari metode *nearest neighbourhood heuristic*

C = Jumlah titik yang ada pada permasalahan

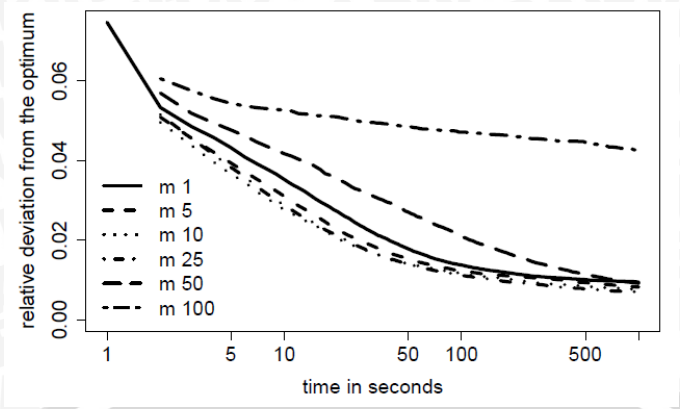
Penetepan parameter dapat berubah sesuai dengan besarnya suatu permasalah yang dihadapi kecuali unguk nilai t_0 dengan menggunakan persamaan 2.5. Jumlah semut yang digunakan untuk mendapatkan hasil terbaik pada algoritma *Ant Colony Optimization* adalah M = 10. (Leksono, 2009)

Hasil nilai q0 dan M ditunjukkan pada gambar 2.9 dan gambar 2.10



Gambar 2.9 Perbandingan hasil Algoritma *Ant Colony Optimization* dengan nilai dari q0

Sumber : Stutzle, Lopez-Ibanez, Pellegrini, Maur, Montes De Oca, Birattari, & Dorigo (2010)



Gambar 2.10 Perbandingan hasil Algoritma *Ant Colony Optimization* dengan nilai dari jumlah semut

Sumber : Stutzle, Lopez-Ibanez, Pellegrini, Maur, Montes De Oca, Birattari, & Dorigo (2010)

2.7 Algoritma *Brute Force*

Algoritma *Brute Force* adalah sebuah cara pendekatan langsung (*straight forward*) untuk memecahkan masalah, yang biasanya didasarkan pada pernyataan masalah (*problem statement*) dan definisi konsep akan dilibatkan. Algoritma *brute force* merupakan alur penyelesaian suatu masalah dengan cara berpikir sederhana dan tidak membutuhkan pemikiran yang lama. Pada dasarnya algoritma *brute force* merupakan algoritma yang memiliki pola pikir manusia yaitu langsung (*to the point*).

Karakteristik dari algoritma *brute force* sebagai berikut (Munir, 2010):

1. Membutuhkan jumlah langkah yang banyak dalam menyelesaikan suatu masalah sehingga jika diterapkan dalam algoritma program aplikasi maka akan membutuhkan banyak memori.
2. Digunakan sebagai dasar dalam menemukan suatu solusi yang lebih efektif.
3. Banyak dipilih dalam penyelesaian sebuah masalah yang sederhana karena sifatnya yang langsung (*straight forward*).
4. Pada banyak kasus, algoritma ini banyak dipilih karena hampir dapat dipastikan dapat menyelesaikan persoalan yang ada.
5. Digunakan sebagai dasar bagi perbandingan keefektifan sebuah algoritma

Algoritma *brute force* memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas (*obvious way*). Algoritma ini secara langsung ke pusat permasalahan. Algoritma ini juga biasanya tidak perlu memerlukan teori khusus untuk mengimplementasikannya. Algoritma *brute force* juga disebut

algoritma sapu jagad karena dapat menyelesaikan hampir semua persoalan pemrograman.

2.8 *Waypoint*

Waypoint adalah koordinat yang mengidentifikasi titik di dalam ruang fisik. Koordinat yang digunakan bermacam-macam, tergantung pada aplikasi yang akan diterapkan. Untuk navigasi darat, koordinat berupa bujur dan lintang. Sedangkan untuk navigasi udara, ditambahakan dengan ketinggian. *Waypoint* digunakan untuk sistem navigasi pada GPS dan berbagai jenis dari radio navigasi. (Sirenden & Dachi, 2012)

Waypoint yang terletak pada permukaan bumi dapat didefinisikan dalam dua dimensi (bujur dan lintang), namun untuk atmosfer bumi atau di luar angkasa akan didefinisikan setidaknya tiga atau empat dimensi jika waktu termasuk salah satu koordinat untuk beberapa titik yang berada di luar bumi. *Waypoint* digunakan untuk membantu untuk menemukan jalur rute navigasi yang tidak terlihat. Misalnya, *artificial airways " highways in the sky "* yang dibuat khusus untuk keperluasn navigasi udara dan hanya terdiri dari serangkaian *waypoint* abstrak di langit yang bertujuan untuk memudahkan pilot dalam mengkontrol lalu lintas udara dan rute lalu lintas pada saat melakukan perjalanan. *Waypoint* abstrak di buat secara praktis dengan teknologi navigasi modern. Seperti *land-based radio beacons* dan satelit berbasis GPS. (Sirenden & Dachi, 2012)

2.8.1 Perhitungan Jarak Antara Dua *Waypoints*

Melakukan perhitungan jarak antara dua *waypoints* pada *Google Maps* dengan menggunakan persamaan hukum kosinus koordinat bola (*spherical law of cosines*) (Sirenden & Dachi, 2012)

$$D = Acos(\sin(\lambda_1) * \sin(\lambda_2) + cos(\lambda_1) * cos(\lambda_2) * \cos(\varphi_2 - \varphi_1) * R \quad (2.6)$$

Dimana :

D : Jarak antar dua titik (km)

λ_1 : Posisi titik i(1,2....n) pada garis lintang dalam derajat

φ_1 : Posisi titik i(1,2....n) pada garis bujur dalam derajat

R : Jari-jari bumi (6371 km)

Dalam menggunakan persamaan ini, nilai lintang dan bujur harus dalam kondisi radian, sedangkan data masukkan dari pengguna dalam kondisi derajat. Maka harus mengubah nilah derajat menjadi radian dengan membagi dengan 180 / π atau 57.29578 (Sirenden & Dachi, 2012)

2.9 *Global Positioning System* (GPS)

Global Positioning System (GPS) adalah sebuah sistem navigasi yang berdasarkan satelit yang saling berhubungan di orbitnya. Satelit-satelit itu diluncurkan pertama kali pada tahun 1978 oleh Departemen Pertahanan (*Departemen of Defense*) Amerika Serikat dan pada tahun 1994 sudah memakai 24 satelit. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yaitu *GPS receiver* yang berfungsi untuk menerima sinyal dari satelit GPS (Prahasta, 2005). Posisi di ubah menjadi titik yang dikenal sebagai *Waypoint* yang nantinya akan berupa titik-titik koordinat lintang dan bujur dari posisi seseorang atau suatu lokasi kemudian ditampilkan pada layar sebagai peta elektronik.

2.9.1 Fungsi GPS

Sistem ini didesain untuk memberikan posisi dan kecepatan menampilkan tiga-dimensi serta informasi mengenai waktu, secara berlanjut di seluruh dunia tanpa bergantung pada waktu dan cuaca secara cepat, akurat, dan murah. (Prahasta, 2005)

2.9.2 Ketelitian GPS

Ketelitian dari GPS mencapai beberapa mm untuk ketelitian posisinya, beberapa cm/s untuk ketelitian kecepatannya dan beberapa nanodetik untuk ketelitian waktunya. Ketelitian posisi yang diperoleh akan tergantung pada beberapa faktor yaitu metode penentuan posisi, geometri satelit, tingkat ketelitian data, dan metode pengolahan datanya. (Prahasta, 2005)

2.9.3 Cara Kerja GPS

Setiap daerah di atas permukaan bumi minimal akan terjangkau oleh 3-4 satelit. Pada kenyataannya, setiap GPS terbaru bisa menerima sampai dengan 12 saluran satelit sekaligus. Kondisi langit yang cerah dan bebas dari halangan akan mempermudah GPS dalam menagkap sinyal yang dikirimkan oleh satelit. Semakin banyak satelit yang diterima oleh GPS, maka akurasi yang diberikan akan semakin tinggi. Logikanya, Cara kerja GPS ada 5 langkah, yaitu (Prahasta, 2005) :

1. Memakai perhitungan secara *Triangulation* dari satelit.
2. Untuk perhitungan *Trigulation*, GPS mengukur jarak menggunakan travel time sinyal radio.
3. Untuk mengukur *Travel Time*, GPS memerlukan akurasi waktu sangat tinggi.
4. Untuk perhitungan jarak, harus mengetahui terlebih dahulu posisi satelit dan ketinggian pada orbitnya secara tepat.

5. Terakhir harus mengoreksi delay sinyal waktu perjalanan di atmosfer sampai diterima oleh *Receiver*.

Satelit GPS berputar mengelilingi bumi selama 12 jam di dalam orbitnya secara konstanta dan mengirimkan sinyal informasi ke bumi. GPS *Receiver* kemudian mengambil informasi dan dengan menggunakan perhitungan *Triangulation* menghitung lokasi penerima dengan tepat. GPS *Receiver* juga membandingkan waktu sinyal di kirim dengan waktu sinyal pada saat telah diterima. Dari informasi itu didapat berapa jarak satelit tersebut. Dengan perhitungan jarak GPS *Receiver* dapat melakukan perhitungan dan menentukan posisi penerima dan menampilkan dalam sebuah peta elektronik. (Prahasta, 2005)

Sebuah GPS *Receiver* harus mengunci sinyal minimal tiga satelit untuk menghitung posisi 2D (*Latitude* dan *Longitude*) serta jalur pergerakannya. Jika GPS *Receiver* dapat menerima empat atau lebih banyak satelit, maka dapat melakukan perhitungan posisi 3D (*Latitude*, *Longitude* dan *Altitude*). Jika sudah dapat menentukan posisi penerima, selanjutnya GPS dapat menghitung informasi lain, seperti kecepatan, arah yang dituju, jalur, tujuan perjalanan, jarak tujuan, matahari terbit dan tenggelam, dan masih banyak lagi. Satelit GPS dalam mengirim informasi waktu sangat tepat karena satelit telah memakai Jam Atom. Pada dasarnya Jam Atom menggunakan standart frekuensi resonansi atom sebagai perhitungannya. Sehingga satelit menghasilkan jam yang sangat akurat dengan keakuratan sebesar 10^{-9} detik per hari. Dengan keakuratannya, satelit dapat menentukan lokasi dari penerima. Selain itu semakin banyak sinyal satelit yang diterima maka akan semakin tepat data yang diterima karena satelit-satelit akan mengirim *Pseudo-Random Code* dalam waktu yang sama. Ketinggian itu menimbulkan keuntungan dalam mendukung proses kerja GPS, Karena semakin tinggi maka semakin bersih Atmosfer. Sehingga gangguan menjadi sedikit serta ketepatan posisi orbit dan perhitungan yang benar. (Prahasta, 2005)

Satelit juga harus berada pada posisi yang benar sehingga stasiun di bumi dapat memonitor setiap pergerakan satelit, dengan bantuan radar yang selalu mengecek *Latitude, Longitude, Altitude, position,* dan kecepatan dari satelit. (Prahasta, 2005)

2.9.4 *Assisted-Global Positioning System* (A-GPS)

Assisted-Global Positioning System (A-GPS) merupakan penyempurnaan dari GPS sebagai penentu posisi di belahan bumi. Satelit GPS di bumi mempunyai 24 susunan satelit dalam enam orbit yang mendekati lingkaran. Setiap orbit ditempati oleh empat buah satelit dengan interval yang berbeda. Orbit satelit GPS berinklinasi 550° terhadap bidang equator dengan ketinggian rata-rata dari permukaan bumi sekitar 20.200 km (El-Rabbany, 2006).

Metode *Advanced Positioning* yang terdapat pada A-GPS merupakan metode penentuan posisi yang paling tinggi akurasinya dibandingkan metode deteksi posisi lainnya seperti *Time Difference Of Arrival* (TDOA) maupun *Enhanced Observed Time Difference* (E-OTD). A-GPS jauh lebih efisien dan efektif dalam

mengakses informasi dari satelit karena tidak perlu mencari data satu persatu dari keseluruhan satelit yang ada, namun A-GPS telah mengetahui target (Satelit) yang dibutuhkan atau dituju (El-Rabbany, 2006)

Pada sistem A-GPS, perangkat bergerak akan menangkap sinyal satelit kemudian dikirimkan ke *Server* penyedia layanan telepon, selanjutnya hasil perhitungan lokasi akan di kirimkan kembali ke perangkat bergerak. Peta juga dapat diunduh terlebih dahulu dari *Server* atau sudah disimpan pada perangkat bergerak. Sistem ini hanya berfungsi bila jaringan perangkat bergerak mampu disediakan oleh penerima. Proses perhitungan data akan dilakukan oleh *Server* penyedia jaringan telepon dan di kirimkan ke penerima (El-Rabbany, 2006).

2.10 Google Maps Android API

Google Maps Android API merupakan layanan pemetaan yang berdasarkan data dari server *Google Maps* kemudian disambungkan ke perangkat bergerak. API (*Application Programming Interface*) secara otomatis akan mengakses ke server *Google Maps* untuk mengunduh data, menampilkan peta, dan melakukan pemetaan bergerak secara *Real-Time*. API ini juga mendukung aksesibilitas dan secara otomatis dapat diaktifkan pada aplikasi yang menggunakan API tersebut. *Google Maps Android API* memiliki 4 jenis pilihan model peta, yaitu : (Sirenden & Dachi, 2012)

1. *ROADMAP*, menampilkan peta dalam bentuk 2 dimensi.
2. *SATELITE*, menampilkan peta dalam bentuk foto dari satelit.
3. *TERRAIN*, menampilkan peta dalam bentuk relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi. Seperti menunjukkan gunung.
4. *HYBRID*, menampilkan peta dalam bentuk foto satelit namun diatasnya tergambar pula apa yang ditampilkan dalam peta *ROADMAP* yaitu jalan dan nama kota.

2.11 Sistem Operasi Android

Android adalah sebuah sistem operasi mobile yang berbasiskan sistem operasi linux. Perkembangan Android dimulai dari versi 1.5 (*Cupcake*) hingga saat ini telah mencapai versi 5.0 (*Lollipop*) dan akan terus berlanjut. *Software Development Kit* (SDK) Android menyediakan alat dan *Application Programming Interface* (API) yang diperlukan untuk mengembangkan aplikasi pada sistem operasi Android dengan menggunakan bahasa pemrograman JAVA. (Wahana komputer, 2013)

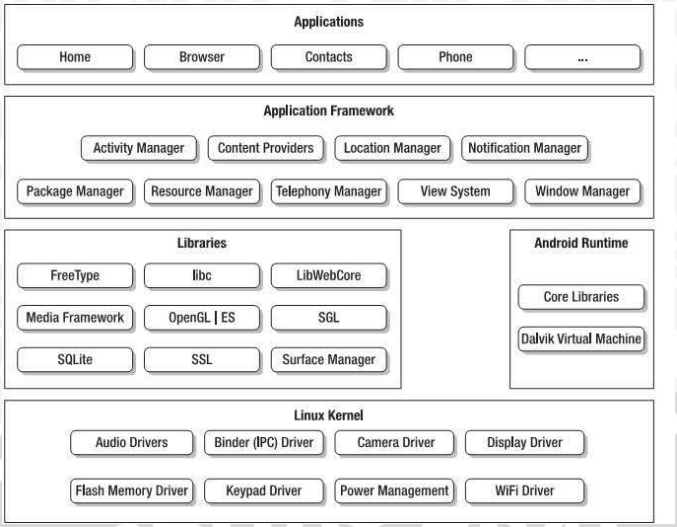
2.11.1 Fitur Android

Android merupakan sebuah sistem operasi perangkat bergerak *Open Source* dan dapat dimanufaktur untuk dikustomisasi sehingga tidak ada konfigurasi yang pasti mengenai *Software* dan *Hardware*-nya. Namun secara garis besar Android sendiri mendukung fitur-fitur berikut ini (Wahana komputer, 2013):

1. *Storage* : Mendukung *SQL Lite*. *SQL Lite* adalah sebuah *Database Relational Lite* (Versi ringan) yang digunakan untuk menyimpan data.
2. Konektivitas : Mendukung koneksi GSM/EDGE, HSDPA, IDEN, CDMA, EVDO, UMTS, Bluetooth, Wifi, LTE, Wimax, dan NFC.
3. *Messaging* : Mendukung SMS dan MMS
4. *Web Browser* : *Web Browser* yang digunakan adalah *Browser* berbasisis *Open Source Webkit*, dengan *Engine Java Script Chrome V8*.
5. *Media Support* : Dukungan mendia meliputi file media bertipe : H.263, H.264 (dalam bentuk 3GP dan MP4 container), MPEG-4 SP, AMR-WB (3GP container), AAC, HE-AAC (dalam bentuk MP4 atau 3GP container), MP3, MIDI, OGG, Vorbis, WAV, JPEG, PNG, GIF, dan BMP
6. Dukungan *Hardware* : Sensor *Accelerometer, Direction, G-Sensor, Light, E-Compass, Gyro, Rotation Vector,* GPS (*Global Positioning System*) dan kamera.
7. *MultiTouch* : Mendukung layar dengan dukungan 2 bahkan 10 sentuhan.
8. *Multitasking* : Kemampuan untuk melaksanaan tugas secara bersamaan.
9. *Tethering* : Mendukung berbagi koneksi internet.
10. *Rich Development Environment* : menjadikan android mempunyai alat pengembangan termasuk *Device Emulator, Debugging, Memory* dan *Performance Profiling,* serta *Plugin* untuk Android Studio.

2.11.2 Arsitektur Android

Untuk lebih memahami mengenai bagaimana arsitektur Android, dapat melihat pada gambar 2.11 :



Gambar 2.11 Arsitektur Android

Sumber : Wahana komputer, 2013

Sesuai gambar 2.11. Arsitektur Android dapat dibagi menjadi lima bagian, yaitu (Wahana komputer, 2013) :

1. *Applications*

Android mempunyai aplikasi inti mencakup *Email Client*, SMS, Kalender, peta, *Browser*, kontak, dan sebagainya. Karena android memiliki lisensi bebas jadi setiap vendor perangkat bergerak dapat memberikan sentuhan kostumisasi yang menjadikan android mempunyai ciri khas dari setiap vendor itu sendiri. Semua aplikasi dibuat dengan bahasa pemrograman JAVA.

2. *Application Framework*

Android menawarkan kemampuan untuk mengembangkan aplikasi yang sangat beragam dan inovatif. Pengembang bebas untuk mengakses perangkat bergerak, informasi akses lokasi, menjalankan *Background Services*, mengatur alarm, mengatur notifikasi, dan lain-lain.

Pengembang juga memiliki akses penuh ke *Framework* API untuk dirancang menggunakan kembali komponen, aplikasi apapun dapat menggunakan kemampuan aplikasi lainnya atau kemampuan aplikasi itu sendiri karena aplikasi tersebut dapat diatur oleh batasan keamanan yang diberlakukan oleh *Framework*. Mekanisme ini mengizinkan pengguna untuk mengganti komponen. Contoh layanan dan sistem yang mendasari seluruh aplikasi Android, yaitu :

1. *View* : Tampilan yang digunakan untuk membuat aplikasi, termasuk didalamnya terdapat *List, Grid,* kotak teks, tombol, dan masih banyak lagi.
2. *Content Providers* : Memungkinkan sebuah aplikasi dapat mengakses data dari aplikasi lainnya (misal Kontak, Mp3, Album), atau membagi data yang dimiliki.

3. *Resource Manager* : Menyediakan akses ke *Non-code Resource* seperti grafik dan *Layout* data.
4. *Notification Manager :* Memungkinkan aplikasi untuk menampilkan pemberitahuan pada *Status Bar*.
5. *Activity Manager* : Mengatur *Lifecycle* aplikasi.

3. *Libraries*

Libraries merupakan *Layer* dimana semua fitur android berada, Pengembang dapat mengakses *Libraries* untuk menjalan aplikasi. Android mempunyai se *Library* C/C++ yang digunakan oleh bermacam-macam komponen dari sistem Android. Beberapa *Library* yaitu :

1. Sistem C *Library* : *Library* standar C (libc).
2. Media *Libraries* : *Library* untuk mendukung memutar dan merekam berbagai format *Audio*, *Video* dan file gambar.
3. *Surface Manager* : Mengatur akses ke tampilan subsistem dan mengabungkan grafik 2D dan 3D pada beberapa aplikasi.
4. *LibWeb Core* : *Web Browser* engine yang mendukung Android *Browser*.
5. SGL : Bagian utama *Engine* grafik 2D
6. 3D *Libraries* : Implementasi *OpenGL ES 1.0 APIs* yang berfungsi untuk mengoptimalkan fungsionalitas 3D.
7. *FreeType* : *Rendering Bitmap* dan *Font*.
8. *SQLite* : merupakan sebuah *Database Relational Lite* (Versi ringan) yang digunakan untuk menyimpan data.

4. Android Runtime

Android Mempunyai seperangkat *Library* inti yang menyediakan sebagian besar fungsi yang tersedia dalam bahasa pemrogram JAVA. Setiap aplikasi Android akan berjalan dalam prosesnya sendiri, dengan *Dalvik Virtual Machine* (DVM). DVM akan mengeksekusi file dalam format *Dalvik Excetubale* (.dex) yang dioptimalkan untuk meminimalisir bekas aplikasi di memori. Namun dalam Android versi 5.0 DVM akan digantikan oleh *Android Runtime* (ART). ART ini memiliki keunggulan melakukan kompilasi data aplikasi telah dilakukan sebelum aplikasi dijalankan dan mengubahnya menjadi aplikasi *Native* di Android. Berkat ART ini memungkinkan Android tidak memerlukan bantuan dari sebuah mesin virtual atau juga menerjemahkan ulang setiap kode pemrograman sehingga mampu mempersingkat proses *Startup* jauh lebih cepat daripada menggunakan DVM.

5. *Linux Kernel*

Android juga bergantung pada *Linux* versi 2.6 untuk layanan sistem inti seperti keamanan, manajemen memori, manajemen proses, jaringan *Stack*, dan *Driver Model.*

2.11.3 Fundamental Aplikasi

Aplikasi Android ditulis dalam bahasa pemrograman JAVA. Kode java dikompilasi bersama dengan data file *Resource* yang dibutuhkan oleh aplikasi , dimana proses di-*Package* oleh alat yang dinamakan *APT Tools* ke dalam paket Android sehingga menghasilkan file dengan ekstensi .apk. file apk inilah sebagai aplikasi Android dan dapat dipasang ke dalam perangkat bergerak. Ada enam jenis komponen pada aplikasi Android, yaitu (Safaat, 2013) :

1. *Activities*

Activities akan menyajikan *User Interface* (UI) pada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi Android bisa jadi hanya memiliki satu *Activity*, tetapi pada dasarnya aplikasi bisa banyak memiliki *Activity,* tergantung pada tujuan dan desain dari aplikasi tersebut. Satu *Activity* biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai UI saat aplikasi diperlihatkan pada pengguna. Secara hirarki sebuah windows *Activity* dinyatakan dengan method *Activity.setContentView()*. *ContentView* adalah objek yang berada pada *root*.

2. *Services*

Service tidak memiliki *Graphic User Interface* (GUI), tetapi *Service* berjalan secara *Background.* Seperti pada aplikasi musik, *Service* mungkin akan memainkan musik atau data jaringan. Namun *Service* harus berada dalam kelas induknya, jadi aplikasi ini akan memiliki dua atau lebih *Activity* yang memungkinkan pengguna untuk memilih lagu seperti mengganti lagu melalui *Status Bar*.

3. *Broadcast Receiver*

Broadcast Receiver berfungsi sebagai menerima dan bereaksi untuk menyampaikan pemberitahuan. Seperti *Broadcast* pemberitahuan zona waktu berubah, pesan baru, baterai *Low,* dan sebagainya. *Broadcast Receiver* tidak memiliki UI, tetapi memiliki *Activity* untuk informasi yang akan diterima.

4. *Content Provider*

Content Provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lainnya. Data akan disimpan dalam *File* sistem seperti *SQLite*. *Content Provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *Activity*.

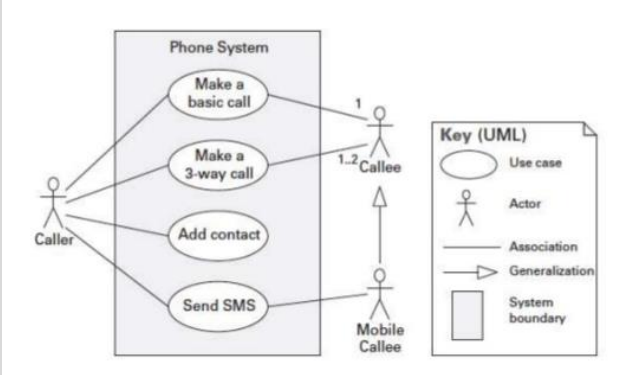
2.12 Unified Modeling Language

Unified Modeling Language (UML) adalah sebuah bahasa standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak. UML juga digunakan untuk merancang model sebuah sistem perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat lunak, sistem operasi, dan sebagainya serta dapat ditulis dengan semua bahasa pemrograman namun kebanyakan dipakai menggunakan bahasa berorientasi objek seperti C++, Java, atau VB.NET. Dapat juga digunakan pada aplikasi yang bersifat prosedular seperti VB atau C. (Prayitno, 2005)

UML juga memiliki notasi dan sematik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram perangkat lunak. Setiap bentuk memiliki arti tertentu, dan UML mengartikan bagaimana bentuk-bentuk tersebut dapat digabungkan. Notasi UML juga diturunkan dari 3 notasi yang telah ada sebelumya yaitu *Grady Booch OOD (Object Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique,* dan *Ivar Jacobson OOSE (Object Oriented Software Engineering)*. Dalam pemodelan, UML dapat digambarkan dengan diagram dimana memiliki kegunaan berbeda. Beberapa diagram diantaranya adalah (Prayitno, 2005):

2.12.1 Use Case Diagram

Diagram *Use Case* adalah diagram yang menunjukkan satu set *use case* dan actor serta hubungan mereka (Prayitno, 2005).



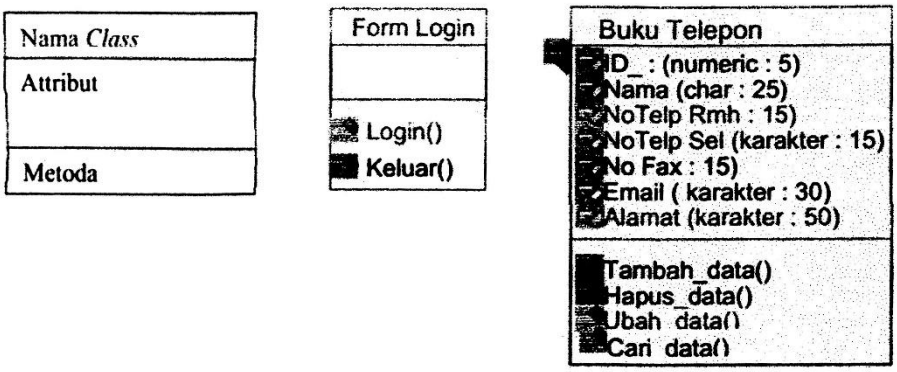
Gambar 2.12 Contoh *Use Case Diagram*

Sumber : Prayitno (2005)

2.12.2 Class Diagram

Class diagram menunjukkan suatu set kelas, antarmuka, kolaborasi dan hubungan antar kelas. Diagram ini paling umum digunakan dalam pemodelan

sistem berorientasi objek untuk menampilkan desain statis dari sebuah sistem dan dapat mengatasi pandangan proses statis dari sebuah sistem (Prayitno, 2005).

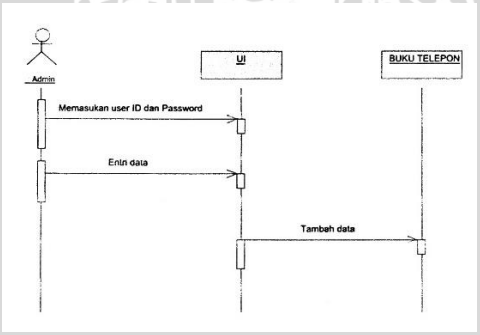


Gambar 2.13 Contoh *Class Diagram*

Sumber : Prayitno (2005)

2.12.3 Sequence Diagram

Diagram *Sequence* adalah diagram yang menggambarkan interaksi antar objek dan mengindikasikan komunikasi diantara objek-objek tersebut. Pada diagram ini, menunjukkan benda diatur sepanjang sumbu X dan pesan, dan perintah waktu sepanjang sumbu Y. Pesan yang dipertukarkan antar objek digambarkan sebagai anak panah antara *Activation Box* pengirim dan penerima. Kemudian atasnya terdapat label pesan. Salah satu contoh *Sequence* diagram digambarkan sebagai berikut (Prayitno, 2005).



Gambar 2.14 Contoh *Sequence Diagram*

Sumber: Prayitno, (2005)

2.12.4 Activity Diagram

Activity diagram atau aktifitas diagram berfungsi untuk menggambarkan *workflow* (aliran kerja) atau sebuah aktifitas sistem atau proses bisnis yang

berguna untuk mendeskripsikan alur kegiatan yang lainnya seperti *use case* atau interaksi. *Activity diagram* juga banyak digunakan untuk mendefinisikan hal-hal berikut (Prayitno, 2005) :

1. Rancangan sebuah proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan alur proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem (*user interface*) dimana setiap aktivitas memiliki sebuah rancangan tampilan antarmuka.
3. Rancangan pengujian dimana setiap aktifitas memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan antarmuka menu yang ditampilkan pada perangkat lunak.

2.13 Konsep Teknik Dan Strategi Pengujian

Ada beberapa cara dan strategi dalam pengujian perangkat lunak, dan semuanya memiliki satu tujuan yang sama, yaitu meningkatkan kepercayaan pengembang terhadap fungsi-fungsi perangkat lunak yang dibuat. Suatu perangkat lunak dapat diuji untuk menerima berbagai perlakuan dalam rangka pengujian. Diantaranya adalah dengan melakukan tindakan-tindakan yang mungkin dilakukan pengguna perangkat lunak dan juga perhitungan terhadap kehandalan perangkat lunak selama operasi dilakukan (Simarmata,2010).

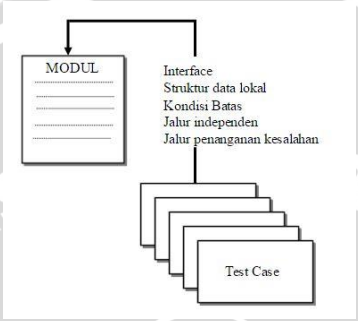
Apapun keterbatasan yang dihadapi, pengujian merupakan bagian yang integral dalam pengembangan perangkat lunak. Pengujian secara luas digunakan di setiap tahapan dalam siklus pengembangan perangkat lunak. Lebih dari 50% waktu pengembangan dihabiskan untuk pengujian. Pengujian dilakukan untuk tujuan berikut (Simarmata,2010):

1. Untuk meningkatkan kualitas
2. Untuk verifikasi dan validasi
3. Untuk keandalan estimasi.

2.13.1 Pengujian Unit

Pengujian unit berfokus pada usaha menganalisa pada inti terkecil dari desain perangkat lunak, yakni modul atau fungsi. Dengan menggunakan gambaran dari desain prosedural sebagai panduan maka jalur kontrol yang penting diuji untuk mencari kesalahan di dalam modul tersebut. Kompleksitas relatif dari pengujian dan kesalahan yang dicari akan dibatasi oleh ruang lingkup yang dibangun untuk pengujian unit. Pengujian unit biasanya berorientasi pada metode *white-box* dengan melakukan tahapan pengujian secara paralel untuk model bertingkat (Pressman,2010).

Antarmuka modul diuji untuk memastikan bahwa informasi secara tepat mengalir masuk dan keluar dari inti program. Struktur data lokal diuji untuk memastikan bahwa data yang tersimpan sementara dapat terjaga selama proses di dalam suatu algoritma dijalankan. Kondisi batas diuji untuk memastikan apakah modul beroperasi dengan tepat pada batas yang ditentukan. Semua jalur yang melalui struktur kontrol dijalankan sedikitnya satu kali. Dan akhirnya, jika menemukan kesalahan proses maka dilakukan pembetulan (Pressman,2010). Pengujian unit dapat dilihat pada gambar 2.15



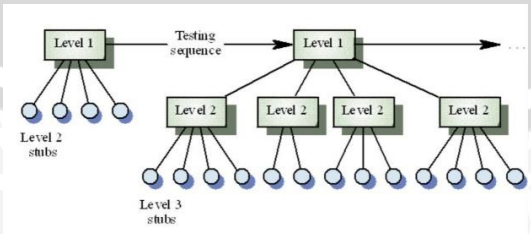
Gambar 2.15 Pengujian Unit

Sumber : Pressman (2010)

2.13.2 Pengujian Integrasi

Pengujian Integrasi adalah sebuah teknik sistematis untuk membangun struktur program serta melakukan pengujian untuk mencari kesalahan yang berhubungan dengan antarmuka. Targetnya adalah untuk mengambil modul yang terhubung dengan pengujian unit dan membangun struktur program yang telah ditentukan oleh desain. Pengujian integrasi memiliki dua pola pengujian, yaitu integrasi *top-down* dan integrasi *bottom-up* (Pressman,2010).

Integrasi *top-down* adalah sebuah pendekatan yang dilakukan secara bertahap terhadap struktur program. Modul diintegrasikan dengan bergerak ke bawah, dimulai dari modul kontrol utama. Subordinat program akan digabungkan ke modul kontrol utama dengan cara *depth-first* atau *breadth-first* (Pressman,2010). Gambar 2.16 menunjukkan pola pengujian integrasi *top-down*.



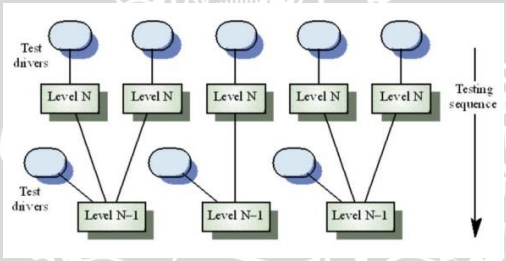
Gambar 2.16 Pengujian Integrasi *Top-down*

Sumber : Pressman (2010)

Proses integrasi *top-down* dilakukan dalam lima langkah, yaitu (Pressman,2010) :

1. Modul kontrol utama digunakan sebagai *test driver* dan *stubs* digunakan untuk menggantikan semua komponen dibawahnya.
2. Pemilihan pendekatan integrasi (*dept first* atau *breadth first*)
3. Pengujian dikerjakan untuk setiap komponen yang terintegrasi
4. *Stubs* digantikan dengan komponen yang sebenarnya setelah menyelesaikan tahapan pengujian.
5. Proses akan terus dilakukan sampai membentuk sebuah perangkat lunak yang utuh.

Pengujian integrasi *bottom-up* memulai pengujian dengan modul atomik (modul tingkat paling rendah pada struktur program). Karena modul diintegrasikan dari bawah ke atas maka alur yang diperlukan untuk modul subordinat ke suatu tingkat yang diberikan akan selalu tersedia dan kebutuhan akan *stubs* dapat dihilangan (Pressman,2010). Gambar 2.17 Menunjukkan pengujian intergrasi *bottom-up*.



Gambar 2.17 Pengujian Integrasi *Bottom-up*

Sumber : Pressman (2010)

Integrasi *bottom-up* dapat diterapkan dengan langkah-langkah berikut (Pressman,2010) :

1. Komponen pada level terendah digabung ke dalam sebuah sub fungsi.
2. *Driver* akan dibuat untuk menguji setiap sub fungsi
3. *Driver* akan diganti dengan modul sesungguhnya setelah sub fungsi telah diuji.
4. Proses akan terus dilakukan sampai membentuk sebuah perangkat lunak.

2.13.3 Pengujian Validasi

Pada pengujian ini dilakukan dengan metode pengujian *blackbox*. Pengujian *blackbox* memfokuskan pada keperluan fungsional dari perangkat lunak. Pengujian ini memungkinkan peneliti untuk membuat himpunan kondisi

input yang akan mencoba seluruh syarat fungsional dari program tersebut. Pengujian ini juga merupakan pendeketan yang berguna untuk menemukan kesalahan lainnya dalam beberapa kategori, diantaranya (Pressman,2010) :

- Fungsi – fungsi yang salah atau hilang
- Kesalahan antarmuka
- Kesalahan dalam struktur data atau akses database
- Kesalahan performa
- Kesalahan inisialisasi dan terminasi.

Pengujian *blackbox* akan dilakukan selama tahap akhr pengujian. Karena pengujian ini memperhatikan struktur control. Alasan peneliti menggunakan pengujian *black box* karena dapat mengetahui apakah perangkat lunak yang akan dibuat dapat berfungsi dengan benar dan telah sesuai dengan diharapkan. Dimana pengujian ini merupakan metode perancangan data uji yang berdasarkan pada spesifikasi perangkat lunak. Nantinya data uji akan dibangkitkan lalu dieksekusi pada perangkat lunak dan kemudian hasil dari perangkat lunak akan dilakukan pengecekan apakah telah sesuai dengan yang diharapkan

2.13.4 Pengujian *Usability*

Pengujian *Usability* dilakukan dalam dua tahap, yaitu tahap pertama pengujian dilakukan untuk menilai seberapa mudah penggunaan aplikasi informasi Walisongo dan tahap kedua pengujian digunakan untuk menilai perspektif hasil pencarian rute yang diperoleh apakah sudah sesuai dengan yang diharapkan oleh peziarah. Dalam pengujian *Usability*, penilaian antarmuka ditargetkan ≥ 83, karena syarat diterimanya sebuah sistem aplikasi dari hasil pengujian *usability* menggunakan skor SUS (*System Usability Scale*) ialah rentang nilai antara 68-100 dapat dilihat pada Gambar 2.18. (Brooke,2013)

System Usability Scale (SUS) merupakan salah satu survey untuk menilai kegunaan dari berbagai produk atau jasa. Definisi dari *usability* menurut standart ISO 9241-11 adalah sejauh mana produk dapat digunakan oleh pengguna tertentu untuk mencapai tujuan tertentu dengan efektifitas, efisiensi, dan kepuasan dalam konteks kegunaan tertentu (Gatsou,2013). SUS memiliki beberapa karakteristik namun yang digunakan dalam sistem aplikasi informasi walisongo ialah dengan mengajukan pertanyaan 10 pertanyaan survey, sehingga dapat dengan mudah dipahami oleh para pengguna untuk memberikan nilai. Perhitungan skor SUS, pertama menjumlahkan skor dari setiap item, setiap item memiliki skor berkisar dari 0 sampai 4. Untuk item bernomor ganjil nilai skor adalah posisi skala yang dipilih responden dikurangi 1. Untuk item bernomor genap nilainya 5 dikurangi posisi skala yang dipilih responden. Lalu kalikan jumlah skor sebesar 2.5 untuk mendapatkan nilai keseluruhan SUS. Skor SUS memiliki kisaran 0 hingga 100 (Brooke,2013). Contoh penggunaan skala SUS ditunjukkan pada tabel 2.1

Tabel 2.1 Skala SUS

No	Pernyataan	Respon					Skor
		1	2	3	4	5	
1	I think that I would like to use this system frequently					X	4
2	I found the system unnecessarily complex				X		1
3	I thought the system was easy to use		X				1
4	I think that I would need the support of a technical person be able to use system	X					4
5	I found the various functions in this system were well integrated		X				1
6	I thought there was too much inconsistency in this system			X			2
7	I would imagine that most people would learn to use this system very quickly		X				1
8	I found the system very cumbersome to use				X		1
9	I felt very confident using the system					X	4
10	I needed to learn a lot of things before I could get going with this system		X				3
Total Skor							**22**

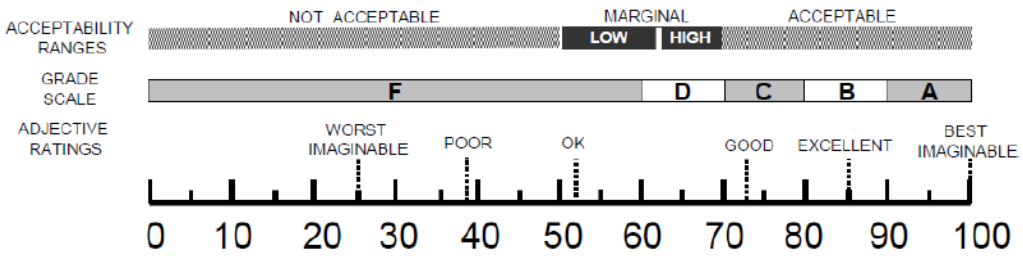
Sumber : Brooke, (2013)

Pilihan Respon :

1. *Strongly Disagree* (Sangat Tidak Setuju)
2. *Disagree* (Tidak Setuju)
3. *Netrall* (Netral)
4. *Agree* (Setuju)
5. *Strongly Agree* (Sangat Setuju)

Nilai SUS Score = 22 * 2.5 = 5.5

Nilai perbandingan hasil skor SUS dengan skor akseptabilitas dan skala penilaian ditunjukkan pada gambar 2.15 (Brooke,2013).



Gambar 2.18 Perbandingan Hasil Skor SUS dengan Skor Akseptabilitas

Sumber : Brooke, (2013)

2.13.5 Pengujian Kompatibilitas

Pada pengujian ini aplikasi diukur dengan menguji di berbagai *platform* perangkat keras yang berbeda seperti versi OS, ukuran layar, resolusi, pengujian terhadap server, serta pengujian aplikasi dalam berinteraksi dengan aplikasi lain pada perangkat keras (Kumar & Chaunan, 2013)

Dengan semua pengujian tersebut nantinya dapat menganalisis dan menghasilkan apakah perangkat lunak sudah memenuhi daftar spesifikasi persyaratan atau belum.

BAB 3 METODOLOGI

Pada bab ini menjelaskan tentang langkah - langkah untuk mengerjakan skripsi ini. Nantinya dalam bab ini akan terdapat sub bab yang akan menjelaskan tentang tipe dan metode penelitian, model proses perangkat lunak, algoritma penelitian, subjek penelitian, lokasi penelitian, perangkat penelitian, Metode pengumpulan data dan strategi pengujian dan analisis hasil pengujian.

3.1 Tipe Dan Metode Penelitian

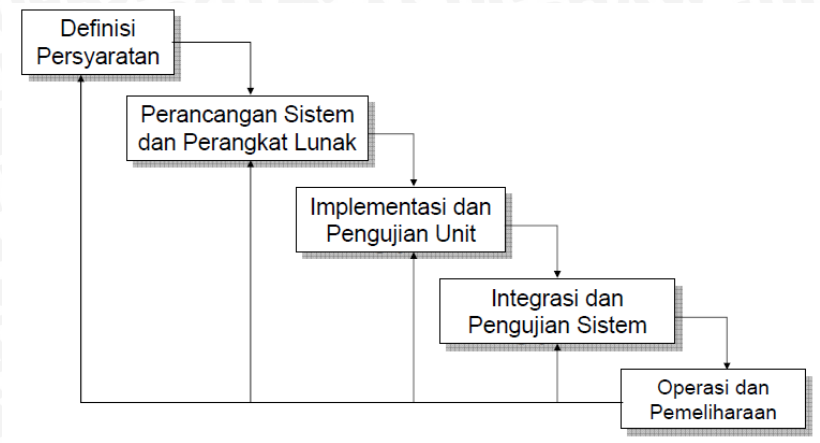
Skripsi ini menggunakan penelitian deskriptif, karena tipe penelitian ini berdasarkan pada pertanyaan bagaimana. Nantinya penemuan dari penelitian deskriptif akan lebih luas dan lebih terperinci karena tidak meneliti masalah itu sendiri. Tetapi juga variabel lain yang berhubungan dengan masalah dalam skripsi ini. Dengan penelitian ini dapat dilakukan dengan menarik sampel. Jadi dapat digunakan untuk mendapatkan hasil yang terbaik (Sugiyono,2008).

Skripsi ini mengunakan metode penelitian *Research and Development* (R&D). Metode ini digunakan untuk menghasil sebuah produk dan akan menguji keefektifan produk tersebut. Produk tersebut tidak selalu berbentuk benda atau perangkat keras. Tetapi dapat pula berbentuk perangkat lunak (Sugiyono,2008)

Tipe dan metode penelitian tersebut dapat berguna dalam penulisan skripsi Rancang Bangun Aplikasi Informasi Walisongo berbasis *Location Based Service* pada *Smartphone Android* karena dalam pelaksanaannya metode penelitian *Research and Development* akan menghasilkan suatu produk perangkat lunak yang akan membantu para peziarah walisongo. Nantinya tipe penelitian deskriptif juga berguna untuk mengembangkan perangkat lunak tersebut menjadi lebih baik.

3.2 Model Proses Perangkat Lunak

Model proses perangkat lunak yang digunakan adalah metode *Waterfall*. Metode ini merupakan suatu proses pengembangan perangkat lunak secara berurutan melewati fase-fase perencanaan, pemodelan, implementasi, dan pengujian. Penulis memilih model ini karena akan lebih mudah di pahami sehingga menjadikan skripsi ini lebih terurut. Berikut adalah gambar metode waterfall (Pressman, 2010) :



Gambar 3.1 Model Proses Perangkat Lunak

1. Definisi Persyaratan

Pada tahap ini akan menganalisis kebutuhan sistem. Pengumpulan data dalam tahap ini bisa melakukan sebuah penelitian, wawancara atau dapat membaca study literatur. Tahapan ini akan menghasilkan definisi persyaratan yang berhubungan dengan kebutuhan pengguna dalam pembuatan sistem. Nantinya dokumen ini akan menjadi acuan sistem analisis untuk menterjemahkan kedalam bahasa pemrograman.

2. Perancangan Sistem dan Perangkat Lunak

Proses perancangan akan menterjemahkan definisi persyaratan menjadi sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum diimplementasikan ke dalam koding. Proses ini berfokus pada struktur data, arsitektur perangkat lunak, representasi interface, dan detail algoritmanya. Tahapan ini akan menghasilkan dokumen *software requirement*. Programmer akan menggunakan dokumen ini untuk melakukan proses pembuatan sistemnya.

3. Implementasi dan Pengujian Unit

Dalam tahapan ini dokumen *software requirement* akan diimplementasikan ke dalam sistem dan nantinya akan menghasilkan sebuah perangkat lunak. Setelah itu akan dilakukan pengujian perangkat lunak dengan tujuan untuk menenemukan kesalahan-kesalahan terhadap sistem yang dibuat kemudian akan diperbaiki.

4. Integrasi dan Pengujian Sistem

Tahapan ini merupakan final karena setelah melakukan analisis, design, dan pengkodean maka sistem sudah siap digunakan oleh pengguna

5. Operasi dan Pemeliharaan

Pada tahap ini akan dilakukan pemeliharaan mengikuti perubahan sistem perangkat lunak jika dibutuhkan atau ketika pengguna menemukan sebuah

masalah di dalam perangkat lunak maka nantinya dapat diperbaiki sehingga layak digunakan oleh pengguna.

3.3 Algoritma Yang Digunakan Pada Penelitian

3.3.1 *Ant Colony Optimization*

Pada dasarnya semut dapat menemukan rute terpendek dalam pejalan dari sarang ke tempat-tempat sumber makanan. Koloni semut dapat menemukan dengan mudahnya rute terpendek antara sarang dan sumber makanan berdasarkan jejak kaki pada lintasan yang telah dilewati. Semakin banyak semut yang melalui suatu lintasan maka akan semakin jelas bekas jejak kakinya melalui bau feromon. Hal ini akan menyebabkan lintasan yang di lalui semut dalam jumlah sedikit maka semakin lama akan semakin berkurang kepadatan semut yang melewatinya, bahkan akan tidak dilewati sama sekali. Sebaliknya lintasan yang dilalui semut dalam jumlah banyak semakin lama akan semakin bertambah jumlah semut yang melewati, atau mungkin semua semut akan melewati lintasan tersebut (Dorigo, Maniezzo, & Colorni, 1991). Dalam skripsi ini Algoritma *Ant Colony Optimization* dipilih karena merupakan salah satu algoritma yang dapat menyelesaikan masalah ini. Dibandingkan dengan algoritma Genetika, Jika jumlah kota relatif banyak waktu pencarian algoritma genetika lebih cepat dibandingkan algoritma semut sedangkan untuk jumlah kota relatif kecil waktu pencarian algoritma semut lebih cepat.

3.3.2 Algoritma *Brute Force*

Brute Force adalah sebuah pendekatan yang lempang (*straight forward)* untuk memecahkan masalah, biasanya didasarkan pula pernyataan masalah (*problem statement*) dan definisi konsep yang dilibatkan. Algoritma ini memecahkan masalah dengan sangat sederha, langsung dan dengan cara yang jelas (*obvious way*) (Munir,2010). Dengan algoritma ini akan mempermudah menemukan rute tercepat yang sudah didapatkan dari algoritma *ant colony optimization*.

3.4 Subjek Penelitian

Subjek penelitian merupakan para peziarah Walisongo ketika aplikasi informasi Walisongo sudah jadi untuk mengetahui daftar spesifikasi persyaratan sistem sudah terpenuhi atau belum.

3.5 Lokasi Penelitian

Penelitian dilakukan di Makam Walisongo Sunan Maulana Malik Ibrahim dan Makam Walisongo Sunan Giri yang berada di kota Gresik Jawa timur. Sunan Malik

Ibrahim merupakan wali pertama yang menyebarkan islam di daerah gresik sekaligus ayah dari Sunan Giri yang merupakan wali keenam.

3.6 Perangkat Penelitian

Pada saat penelitian maka dibutuhkan perangkat penelitian untuk menunjang sarana penelitian. Perangkat yang digunakan dalam penelitian ini dibagi menjadi dua bagian, yaitu perangkat keras (*Hardware*) dan perangkat lunak (*Software*). Perangkat keras yang digunaka adalah komputer dan *Smartphone Android*. Sedangkan untuk perangkat lunak adalah kebutuhan sebuah sistem operasi dan perangkat lunak yang mendukung pembuatan aplikasi ini.

Untuk membuat sebuah aplikasi yang dapat berfungsi optimal diperlukan adanya lingkungan perangkat keras dan perangkat lunak sebagai berikut :

1. Perangkat Keras

- *Notebook* dengan Asus K55VM yang mempunyai spesifikasi Intel Core i5 3210M 2.50 Ghz 64 bit, *Harddisk* 1 TB, *DVD-RW Optical Drive*, RAM 4 GB, VGA NVIDIA GeForce GT 630M 2 GB.
- *Smartphone* dengan Sony Xperia Z Ultra yang mempunyai spesifikasi MSM8974 Snapdragon 800, Quad Core 2.2 Ghz Krait 400, Adreno 330, Internal 16 GB, RAM 2 GB, GPS, GSM/HSPA, WiFi, Resolusi 1080 x 1920 pixels.

2. Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah adt bundlewindows-x86 64-20130729 (Eclipse), JDK (Java Development Kit) 7, Adobe Photoshop CC, Notepad ++, dan Genymotion

Sistem operasi yang digunakan adalah Windows 10 Pro 64 bit. Versi Android yang digunakan adalah 5.0.2 Lollipop.

3.7 Metode Pengumpulan Data

Metode pengumpulan data yang digunakan dalam penulisan skripsi ini ada tiga, yaitu :

1. Observasi

Penulis mengumpulkan data dan informasi dengan meninjau dan mengamati secara langsung di lapangan. Dalam hal ini penulis meninjau secara langsung apakah *latitude* dan *longitude* yang terdapat dalam google maps sudah tepat atau ada kesalahan.

2. Studi Pustaka

Penulis mencari, membaca dan mempelajari buku-buku referensi mengenai perancangan aplikasi dengan menggunakan algoritma *Ant Colony Optimization* dan algoritma *Brute Force*, serta mencari informasi apa itu Android, *Travelling Salesman Problem*, *API Google Maps*, dan sejarah Walisongo untuk menunjang pembuatan aplikasi ini. Selain itu peneliti juga mengunduh beberapa *e-book, e-jurnal,* artikel dari media internet sebagai referensi tambahan.

3. Kuesioner

Pada tahap ini, penulis akan bertanya kepada pengguna ketika aplikasi sudah jadi. Pertanyaan yang penulis ajukan berupa pertanyaan tentang apakah aplikasi sudah berjalan dengan baik dan memenuhi daftar spesifikasi persyaratan sehingga nantinya dapat diketahui masalah apa saja yang terjadi dalam aplikasi sebelumnya dirilis secara menyeluruh.

3.8 Strategi Pengujian dan Analisis Hasil Pengujian

Pada tahap ini, sistem sudah menjadi *prototype* dan akan dilakukan pengujian. Pengujian dilakukan pada penelitian ini ditinjau dari pengujian validasi pengujian sistem, dan pengujian hardware. Berikut pembahasannya :

1. Pengujian Unit

Pengujian unit berfokus pada usaha menganalisa pada inti terkecil dari desain perangkat lunak, yakni modul atau fungsi. Dengan menggunakan gambaran dari desain prosedural sebagai panduan maka jalur kontrol yang penting diuji untuk mencari kesalahan di dalam modul tersebut. Kompleksitas relatif dari pengujian dan kesalahan yang dicari akan dibatasi oleh ruang lingkup yang dibangun untuk pengujian unit. Pengujian unit biasanya berorientasi pada metode *white-box* dengan melakukan tahapan pengujian secara paralel untuk model bertingkat (Pressman,2010).

Antarmuka modul diuji untuk memastikan bahwa informasi secara tepat mengalir masuk dan keluar dari inti program. Struktur data lokal diuji untuk memastikan bahwa data yang tersimpan sementara dapat terjaga selama proses di dalam suatu algoritma dijalankan. Kondisi batas diuji untuk memastikan apakah modul beroperasi dengan tepat pada batas yang ditentukan. Semua jalur yang melalui struktur kontrol dijalankan sedikitnya satu kali. Dan akhirnya, jika menemukan kesalahan proses maka dilakukan pembetulan (Pressman,2010).

2. Pengujian Integritas

Pengujian integrasi *bottom-up* memulai pengujian dengan modul atomik (modul tingkat paling rendah pada struktur program). Karena modul diintegrasikan dari bawah ke atas maka alur yang diperlukan untuk modul subordinat ke suatu tingkat yang diberikan akan selalu tersedia dan kebutuhan akan *stubs* dapat dihilangan (Pressman,2010).

Integrasi *bottom-up* dapat diterapkan dengan langkah-langkah berikut (Pressman,2010) :

1. Komponen pada level terendah digabung ke dalam sebuah sub fungsi.
2. *Driver* akan dibuat untuk menguji setiap sub fungsi
3. *Driver* akan diganti dengan modul sesungguhnya setelah sub fungsi telah diuji.
4. Proses akan terus dilakukan sampai membentuk sebuah perangkat lunak.

3. Pengujian Validasi

Pada pengujian ini dilakukan dengan metode pengujian *blackbox*. Pengujian *blackbox* memfokuskan pada keperluan fungsional dari perangkat lunak. Pengujian ini memungkinkan peneliti untuk membuat himpunan kondisi *input* yang akan mencoba seluruh syarat fungsional dari program tersebut. Pengujian ini juga merupakan pendeketan yang berguna untuk menemukan kesalahan lainnya dalam beberapa kategori, diantaranya (Pressman,2010) :

- Fungsi – fungsi yang salah atau hilang
- Kesalahan interface
- Kesalahan dalam struktur data atau akses database
- Kesalahan performa
- Kesalahan inisialisasi dan terminasi.

Pengujian *blackbox* akan dilakukan selama tahap akhr pengujian. Karena pengujian ini memperhatikan struktur control. Alasan peneliti menggunakan pengujian *black box* karena dapat mengetahui apakah perangkat lunak yang akan dibuat dapat berfungsi dengan benar dan telah sesuai dengan diharapkan. Dimana pengujian ini merupakan metode perancangan data uji yang berdasarkan pada spesifikasi perangkat lunak. Nantinya data uji akan dibangkitkan lalu dieksekusi pada perangkat lunak dan kemudian hasil dari perangkat lunak akan dilakukan pengecekan apakah telah sesuai dengan yang diharapkan

4. Pengujian *Usability*

Pengujian *Usability* dilakukan dalam dua tahap, yaitu tahap pertama pengujian dilakukan untuk menilai seberapa mudah penggunaan aplikasi informasi Walisongo dan tahap kedua pengujian digunakan untuk menilai perspektif hasil pencarian rute yang diperoleh apakah sudah sesuai dengan yang diharapkan oleh peziarah. Dalam pengujian *Usability*, penilaian antarmuka ditargetkan ≥ 83 , karena syarat diterimanya sebuah sistem aplikasi dari hasil pengujian *usability* menggunakan skor SUS (*System Usability Scale*).

3. Pengujian Kompatibilitas

Kompatibilitas aplikasi *mobile* dapat diukur dengan menguji di berbagai *platform* perangkat keras yang berbeda seperti versi OS, ukuran layar, resolusi, pengujian terhadap perubahan server, serta pengujian aplikasi dalam berinteraksi dengan aplikasi lain pada perangkat keras

Dengan semua pengujian tersebut nantinya dapat menganalisis dan menghasilkan apakah perangkat lunak sudah memenuhi daftar spesifikasi persyaratan atau belum.

BAB 4 REKAYASA PERSYARATAN

Rekayasa persyaratan adalah proses yang melibatkan semua kegiatan yang akan dibutuhkan untuk membuat dan memelihara dokumen persyaratan sistem. Nantinya pada bab ini akan menjelaskan tentang gambaran umum sistem informasi, identifikasi aktor, analisis data, daftar spesifikasi persyaratan dan pemodelan diagram *use case* yang akan berguna untuk membuat perangkat lunak yang sesuai.

4.1 Gambaran Umum Sistem Informasi

Pada bagian ini, gambaran umum sistem informasi aplikasi Walisongo terdiri dari dua bagian, yaitu deskripsi umum dan lingkungan sistem informasi aplikasi Walisongo.

1. Deskripsi Sistem Informasi Aplikasi Walisongo

Sistem informasi aplikasi Walisongo dapat digunakan pengguna dalam hal ini peziarah untuk mengetahui informasi tentang Walisongo seperti sejarahnya dan tempat lokasi makam walisongo. Dengan dibuatnya aplikasi ini diharapkan juga membantu para peziarah untuk berziarah ke makam walisongo karena terdapat fitur pencarian rute tercepat berdasarkan lokasi awal (peziarah) secara cepat dan semua makam walisongo dapat dikunjungi.

2. Lingkungan Sistem Informasi

Sistem informasi aplikasi Walisongo ini membutuhkan suatu lingkungan yang digunakan untuk berjalanannya sistem. Secara keseluruhan sistem informasi ini berbasis aplikasi *Mobile Hybrid* dimana waktu pembuatannya ditambahkan kode web (HTML dan *JavaScript*) dengan *Software development kit* (SDK). Dan hasil akhirnya adalah sebuah aplikasi mobile.

4.2 Identifikasi Aktor

Identifikasi aktor digunakan untuk mengidentifikasi terhadap aktor-aktor yang akan berinteraksi dengan aplikasi. Tabel 4.1 memperlihatkan aktor-aktor yang terlibat beserta penjelasannya.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
User	User adalah pengguna yang memiliki hak untuk menggunakan fitur mencari rute tercepat, load rute, dan mengetahui tentang informasi walisongo

4.3 Analisis Data

Analisis data berguna untuk mendapatkan struktur penyimpanan data yang dibutuhkan nanti. Struktur penyimpanan data pada aplikasi disusun berdasarkan analisis data sebagai berikut.

1. Data nama-nama Walisongo
2. Data Latitude dan Longitude Makam Walisongo
3. Data lokasi awal pengguna
4. Data Informasi yang terdiri dari foto dan sejarah walisongo

4.4 Daftar Spesifikasi Persyaratan

Daftar spesifikasi persyaratan terdiri dari kebutuhan fungsional dan non-fungsional. Disini peziarah atau user akan dispesifikasikan kebutuhan fungsionalnya seperti pada tabel 4.2 dan non fungsional terdapat pada tabel 4.3

Tabel 4.2 Spesifikasi Kebutuhan Fungsional

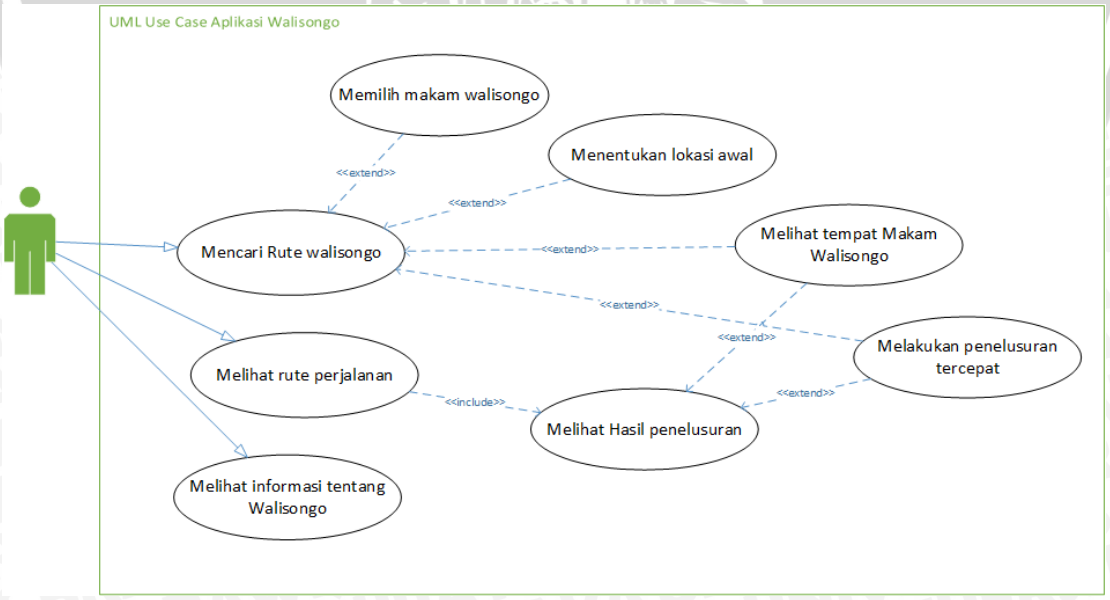
Nomor SRS	Kebutuhan	*Use Case*
SRS_01	Perangkat lunak harus mampu menyediakan fasilitas pencarian rute tercepat dari lokasi awal (user)	Mencari Rute Walisongo
SRS_02	Perangkat lunak dapat menampilkan informasi rute yang di dapat tanpa harus melakukan penelusuran lagi	Melihat rute perjalanan
SRS_03	Perangkat lunak dapat menampilkan informasi mengenai sejarah walisongo beserta foto walisongo itu sendiri	Melihat informasi tentang walisongo

Tabel 4.3 Daftar kebutuhan non-fungsional

Parameter	Kode	Deskripsi Kebutuhan
Usability	1	Aplikasi dapat diakses dengan mudah oleh pengguna
	2	Aplikasi memiliki fitur untuk mengakses load rute dimana nantinya fitur ini akan menampilkan pencarian awal tanpa perlu membuka aplikasi dalam bentuk files .txt
	3	Aplikasi juga memiliki fitur penjelasan tentang sejarah walisongo dalam menu Informasi Walisongo
Compatibility	4	Aplikasi hanya dapat diakses pada *Smartphone* berbasis sistem operasi Android.

4.5 Diagram *Use Case*

Diagram *use case* adalah salah satu diagram untuk memodelkan fungsi yang terdapat dalam sistem. Diagram ini menunjukkan sekumpulan *use case*, aktor, dan hubungannya. Intinya *use case* digunakan untuk menampilkan fungsionalitas dari sistem yang diiinisialisasi oleh aktor. Berikut adalah gambar *use case* sistem yang ditunjukkan pada Gambar 4.1



Gambar 4.1 Diagram *Use Case* Sistem

Pada diagram *use case* di atas melibatkan *user* atau peziarah sebagai aktor Nantinya, delapan buah use case ini akan disertakan dengan skenario *use case* untuk menjelaskan rangkaian aktivitas yang terjadi di *use case* tersebut.

Tabel 4.4 Memilih Makam Walisongo

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_01_1
Nama	Memilih Makam Walisongo
Tujuan	Pengguna dapat memilih makam para walisongo yang akan dikunjungi.
Deskripsi	*Use Case* ini menjelaskan bagaimana pengguna dapat memilih makam para walisongo yang nantinya akan dikunjungi
Aktor	Pengguna (*user*)
Skenario Utama	
Kondisi Awal	Sistem sudah berjalan dan menampilkan halaman menu
Aksi Aktor	**Reaksi Sistem**
1. *User* memilih dan menekan pada pilihan makam walisongo	Sistem akan menampilkan titik awal pengguna
Skenario Alternatif 1 : Jika memilih satu tujuan	
	Penelusuran tidak dapat berjalan
Skenario Alternatif 2 : Jika memilih minimal 3 tujuan	
	Penelusuran dapat berjalan

Kondisi Akhir	Sistem dapat berjalan dan menampilkan skenario selanjutnya yaitu menentukan lokasi awal pengguna.

Tabel 4.5 Menentukan Lokasi Awal

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_01_2
Nama	Menentukan Lokasi Awal
Tujuan	Agar pengguna dapat memastikan lokasi awal.
Deskripsi	*Use Case* ini menjelaskan bagaimana proses menetukan lokasi awal yaitu tempat pengguna saat ini
Aktor	Pengguna (*user*)
Skenario Utama	
Kondisi Awal	Sistem sudah berjalan dan menampilkan halaman menu
Aksi Aktor	**Reaksi Sistem**
2. *User* menekan tombol penelusuran pada Sistem	Sistem akan menampilkan titik awal pengguna
Skenario Alternatif 1 : Jika GPS mati	
	Penelusuran tidak dapat berjalan
Skenario Alternatif 2 : Jika paket data mati	

	Penelusuran tidak dapat berjalan
Kondisi Akhir	Sistem menampilkan lokasi awal pengguna

Tabel 4.6 Melihat Tempat Walisongo

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_01_3
Nama	Melihat Tempat Makam Walisongo
Tujuan	Agar pengguna dapat mengetahui dimana letak makam para Walisongo
Deskripsi	*Use Case* ini menjelaskan bagaimana pengguna dapat mengetahui letak makam para walisongo saat ini
Aktor	Pengguna (*user*)
Skenario Utama	
Kondisi Awal	Sistem sudah berjalan dan menampilkan halaman menu
Aksi Aktor	**Reaksi Sistem**
1. *User* menekan tombol mulai penelusuran pada Sistem	Sistem akan melakukan pemetaan sehingga menampilkan lokasi makam para Walisongo
Skenario Alternatif 1 : Jika GPS mati	
	Penelusuran tidak dapat berjalan
Skenario Alternatif 2 : Jika paket data mati	

	Penelusuran tidak dapat berjalan
Kondisi Akhir	Sistem menampilkan lokasi para makam walisongo beserta lokasi awal pengguna

Tabel 4.7 Melakukan Penelusuran Tercepat

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_01_4
Nama	Melakukan penelusuran tercepat
Tujuan	Agar pengguna dapat mengetahui rute tercepat
Deskripsi	*Use Case* ini menjelaskan bagaimana pengguna dapat mengetahui rute tercepat ke makam para walisongo berdasarkan lokasi awal
Aktor	Pengguna (*user*)
Skenario Utama	
Kondisi Awal	Sistem sudah berjalan dan menampilkan halaman menu
Aksi Aktor	**Reaksi Sistem**
1. *User* menekan tombol mulai penelusuran pada Sistem	Sistem akan melakukan penelusuran sehingga pengguna dapat dengan mudah menentukan makam Walisongo yang akan dikunjungi terlebih dahulu hingga akhirnya semua makam para Walisongo sudah dikunjungi serta rute tercepat akan disimpan kedalam file.txt

Skenario Alternatif 1 : Jika GPS mati	
	Penelusuran tidak dapat berjalan
Skenario Alternatif 2 : Jika paket data mati	
	Penelusuran tidak dapat berjalan
Skenario Alternatif 3 : Jika menekanan tombol rute pulang	
	Sistem akan melakukan penelusuran dari lokasi makam walisongo yang terakhir dikunjungi menuju lokasi awal
Kondisi Akhir	Sistem menampilkan rute tercepat menuju makam para Walisongo dan rute pulang menuju lokasi awal dan menyimpan hasilnya dalam files.txt

Tabel 4.8 Melihat Hasil Penelusuran

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_02
Nama	Melihat Hasil Penelusuran
Tujuan	Agar pengguna dapat mengetahui hasil penelusuran yang telah dicari tanpa melakukan penelusuran lagi
Deskripsi	*Use Case* ini menjelaskan bagaimana menampilkan hasil penelusuran rute tercepat menuju makam para Walisongo dari lokasi awal yang telah disimpan oleh system
Aktor	Pengguna (*user*)

Skenario Utama	
Kondisi Awal	Sistem sudah berjalan dan menampilkan halaman menu
Aksi Aktor	**Reaksi Sistem**
1. *User* menekan tombol load rute	Sistem akan menampilkan hasil rute tercepat kepada pengguna yang telah disimpan oleh system
Kondisi Akhir	Sistem menampilkan hasil rute tercepat yang telah disimpan kepada pengguna

Tabel 4.9 Melihat Informasi Tentang Walisongo

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS_03
Nama	Informasi Tentang Walisongo
Tujuan	Agar pengguna dapat mengetahui foto dan sejarah singkat mengenai Walisongo
Deskripsi	*Use Case* ini menjelaskan bagaimana menampilkan informasi mengenai Walisongo
Aktor	Pengguna (*user*)
Skenario Utama	
Kondisi Awal	Sistem sudah berjalan dan menampilkan halaman menu
Aksi Aktor	**Reaksi Sistem**

2. *User* menekan tombol Informasi Tentang Walisongo	Sistem akan menampilkan halaman pilihan menu nama para Walisongo dari urutan awal
3. *User* menekan tombol nama para Walisongo	Sistem akan menampilkan halaman berisi foto dan sejarah singkat para walisongo yang sudah dipilih oleh pengguna
Kondisi Akhir	Sistem menampilkan halaman berisi foto dan sejarah singkat para walisongo yang sudah dipilih oleh pengguna

BAB 5 PERANCANGAN DAN IMPLEMENTASI

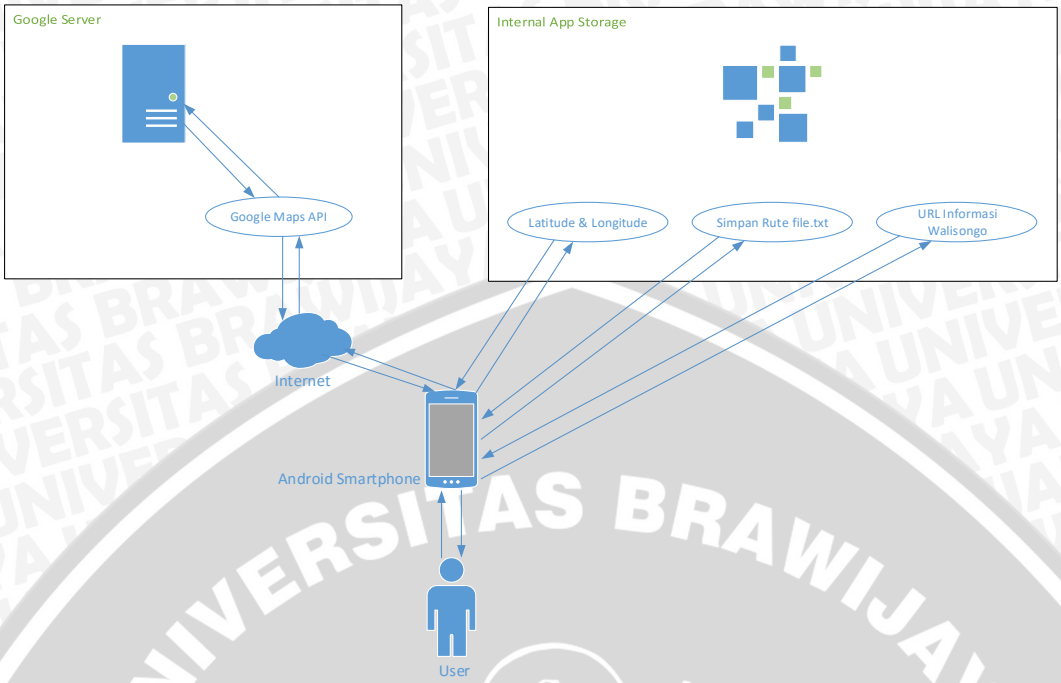
Pada bab ini akan menjelaskan lebih detail mengenai rancangan arsitektur yang berupa deskripsi struktur dan setiap komponen utama. Lalu menjelaskan mengenai representasi data dalam model data serta detil implementasi dari fungsi – fungsi utama yang menjadi fokus dalam pembuatan aplikasi Walisongo.

5.1 Perancangan Aplikasi

Perancangan aplikasi dilakukan dalam enam tahap, yaitu perancangan arsitektural, pemodelan *class diagram* yang berguna untuk menggambarkan perancangan struktur *class* yang menyusun aplikasi walisongo, perancangan *activity diagram*, perancangan *sequence diagram* untuk menggambarkan interaksi antar objek atau *class* di dalam aplikasi, perancangan algoritma berguna untuk menjelaskan algoritma yang akan digunakan dalam pembuatan aplikasi dan perancangan antarmuka pengguna dari aplikasi, perancangan aplikasi pada skripsi ini menggunakan pendekatan desain berorientasi objek yang ditulis dengan menggunakan *Unified Modelling Language* (UML).

5.1.1 Perancangan Arsitektural

Aplikasi informasi walisongo dibangun menggunakan teknologi *hybrid mobile application* yang menggunakan bahasa pemrograman java dan javascript untuk memanggil mode *view web* dari api *google maps* yang nantinya menampilkan peta. Proses kerja sistem dimulai dari pengguna menjalankan aplikasi, dalam tahap ini aplikasi berbentuk *native.* Selanjutnya ketika pengguna menekan tombol penelusuran maka aplikasi akan menampilkan pilihan makam Walisongo yang akan dikunjungi. Setelah memilih tujuan, maka selanjutnya aplikasi akan memanggil mode *web view* yang menampilkan *google maps* dan mencari titik lokasi awal pengguna berdasarkan *GPS* serta jaringan paket data. Selanjutnya aplikasi menampilkan titik aplikasi awal beserta titik makam para walisongo yang koordinatanya telah disimpan dalam aplikasi. lalu aplikasi mencari rute tercepat menggunakan algoritma *ant colony optimization* dan algoritma *brute force.* Selanjutnya ditampilkan dalam *maps* beserta rute tercepatnya. Hasil yang didapatkan dari pencarian akan otomatis disimpan kedalam bentuk berkas txt yang nantinya dapat dipanggil kembali pada menu load rute. Sehingga memudahkan pengguna tanpa perlu membuka menu penelusuran. Menu informasi tentang walisongo menampilkan sejarah singkat dan foto para walisongo yang telah tersimpan dalam berkas *html*. Nantinya aplikasi melakukan pemanggilan berkas html yang berada di dalam aplikasi untuk ditampilkan kedalam mode view web. Desain arsitektural hubungan sistem yang dibangun ditunjukkan pada gambar 5.1

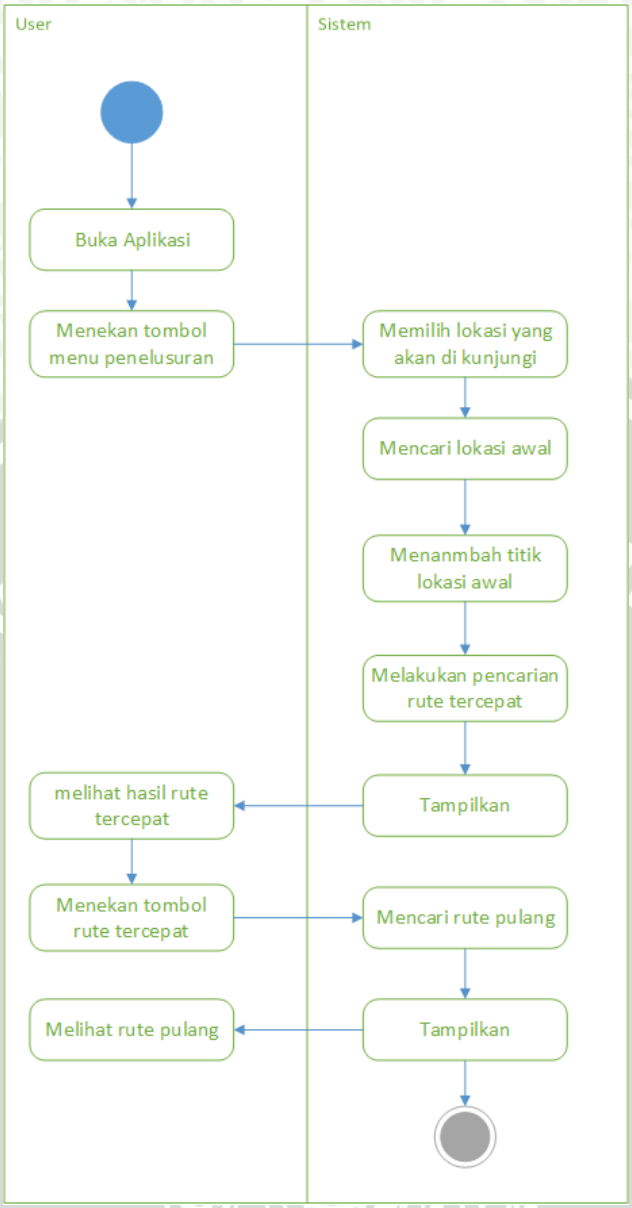


Gambar 5.1 Perancangan Arsitektural Sistem

5.1.2 Perancangan *Activity Diagram*

1. *Activity Diagram* Mencari Rute Walisongo

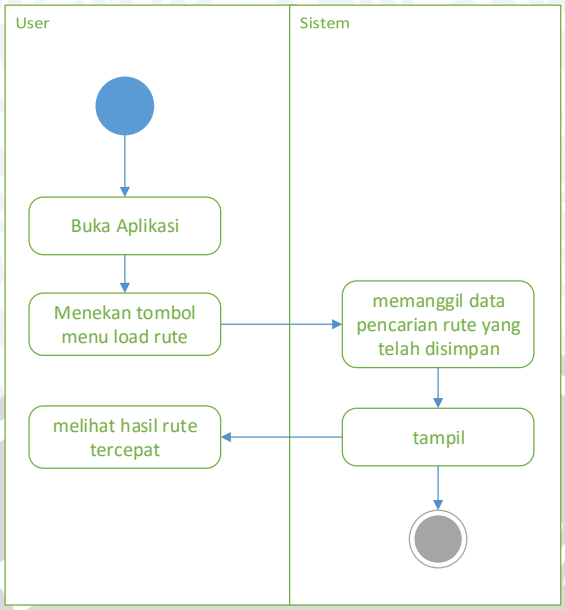
Activity Diagram mencari rute Walisongo adalah kegiatan ketika pengguna ingin mencari rute tercepat menuju ke makam para Walisongo. Perancangan Activity Diagram ditunjukkan pada gambar 5.2.



Gambar 5.2 *Activity Diagram* Mencari Rute Walisongo

2. *Activity Diagram* Melihat Rute Perjalanan

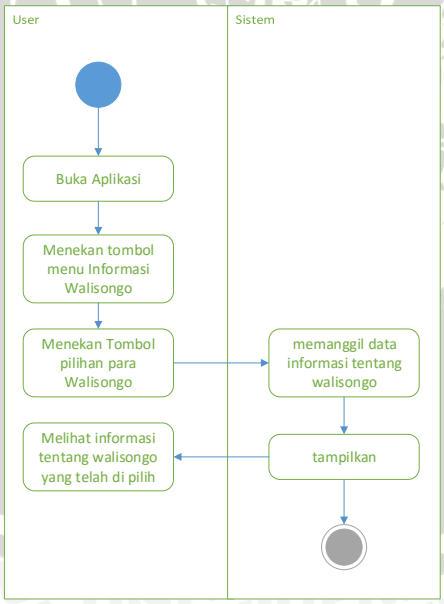
Activity Diagram melihat rute perjalanan adalah kegiatan ketika pengguna ingin melihat hasil penelusuran yang sudah dilakukan terlebih dahulu tanpa perlu melakukan penelusuran lagi. Perancangan *activity diagram* dapat dilihat pada gambar 5.3.



Gambar 5.3 *Activity Diagram* melihat rute perjalanan

3. *Activity Diagram* Melihat Informasi Walisongo

Activity diagram melihat informasi walisongo adalah kegiatan ketika pengguna ingin mengetahui informasi sejarah singkat mengenai para Walisongo. Perancangan *activity diagram* melihat rute walisongo dapat dilihat pada gambar 5.4.

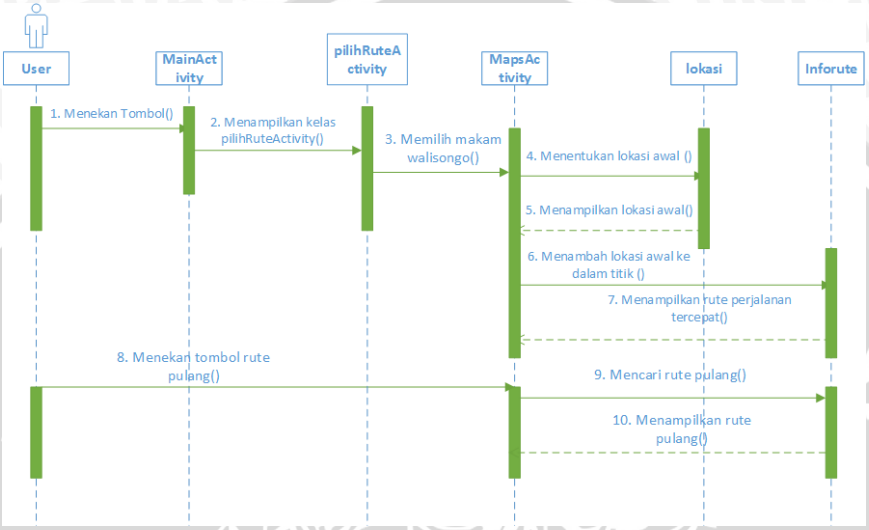


Gambar 5.4 *Activity Diagram* Melihat Informasi Walisongo

5.1.3 Perancangan *Sequence Diagram*

1. *Sequence Diagram* Mencari Rute Walisongo

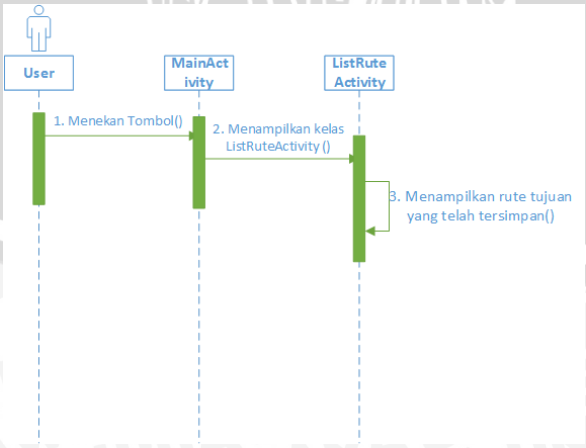
Sequence diagram mencari rute walisongo adalah kegiatan ketika pengguna ingin melakakukan pencarian rute ke makam para Walisongo. Langkah awal yang dilakukan adalah menentukan lokasi awal pengguna sebagai titik keberangkatan dan tujan pulang. Perancangan *sequence diagram* dapat dilihat pada gambar 5.5.



Gambar 5.5 *Sequence Diagram* Mencari Rute Walisongo

2. *Sequence Diagram* Melihat Rute Walisongo

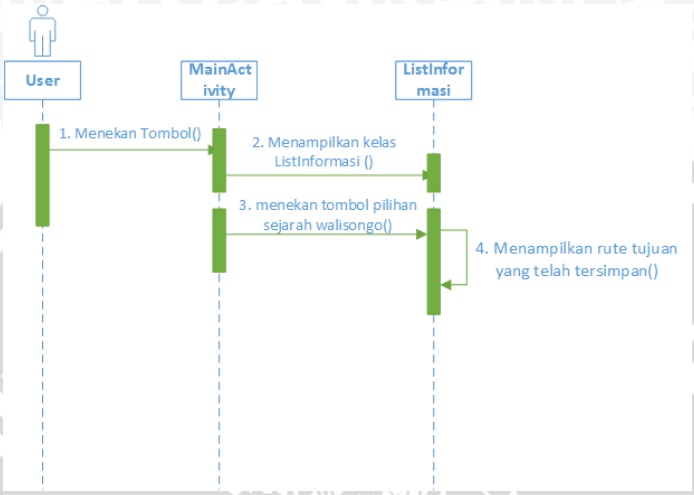
Sequence diagram melihat rute Walisongo adalah kegiatan ketika pengguna ingin melihat rute yang telah di cari sebelumnya. Langkah awal yang harus dilakukan tentunya sistem sudah melakukan kegiatan mencari rute Walisongo. Perancangan *sequence diagram* melihat rute Walisongo dapat dilihat pada gambar 5.6



Gambar 5.6 *Sequence Diagram* Melihat Rute Walisongo

3. *Sequence Diagram* Melihat Informasi Tentang Walisongo

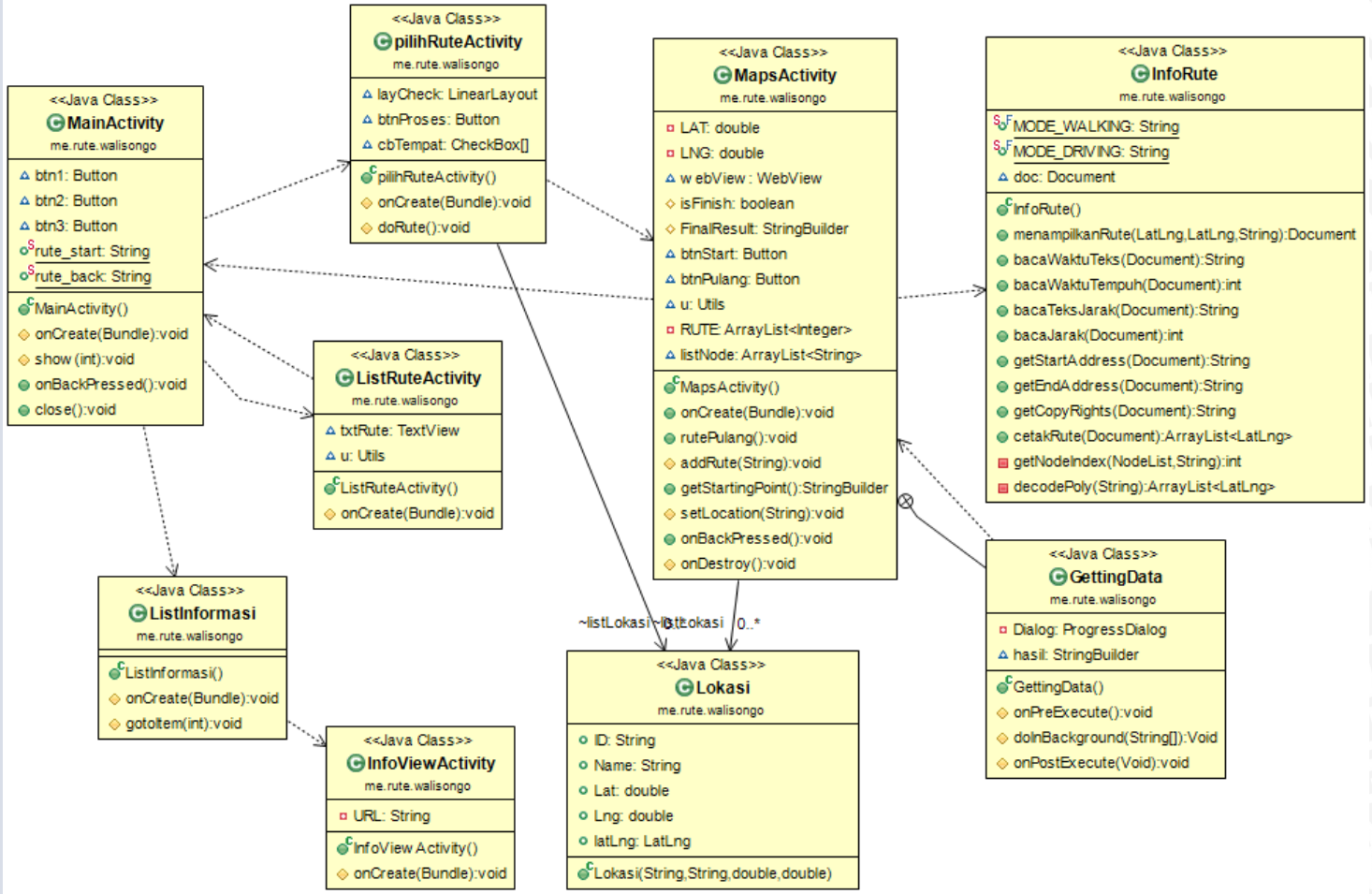
Sequence Diagram melihat informasi tentang Walisongo adalah kegiatan ketika pengguna melihat sejarah singkat mengenai para Walisongo. Perancangan *sequence diagram* melihat informasi informasi tentang Walisongo dapat dilihat pada gambar 5.7.



Gambar 5.7 *Sequence Diagram* Melihat Informasi Tentang Walisongo

5.1.4 Perancangan *Class Diagram*

Class Diagram memberikan gambaran pemodelan elemen *class* yang membentuk sebuah sistem. *Class* bisa didapatkan dengan mengamati secara jelas terhadap *use case* yang dimodelkan. Pada tahap ini, *class diagram* yang dibuat dirancang berdasarkan struktur aplikasi android yang dapat digambarkan pada gambar 5.8



Gambar 5.8 Class Diagram

Pada gambar 5.8 menjelaskan *class diagram* aplikasi informasi walisongo. Pada kelas `MainActivity`, memiliki hubungan *dependecy* (kebergantungan) dengan kelas `pilihRuteActivity`, kelas `ListRuteActivity`, dan kelas `ListInformasi`. Sehingga menunjukkan jika terdapat perubahan pada kelas `MainActivity` maka akan berpengaruh pada kelas `pilihRuteActivity`, kelas `ListRuteActivity`, dan kelas `ListInformasi`.

Kelas `pilihRuteActivity` memilih hubungan *dependency* dengan kelas `MapsActivity`, namun kelas `pilihRuteActivity` juga memiliki hubungan asosiasi dengan kelas lokasi dimana satu kelas digunakan untuk kelas lainnya. Kelas `MapsActivty` memiliki hubungan *dependency* dengan kelas `InfoRute` dan kelas `MainActivity` serta memiliki kelas `GettingData` di dalam kelas `MapsActivity`. Kelas `GettingData` juga memiliki hubungan *dependency* dengan kelas `MapsActivity`. Kelas `ListRuteActivity` memiliki hubungan *dependency* dengan kelas `MainActivity`, untuk kelas `ListInformasi` memeliki hubungan *dependency* dengan kelas `InfoViewActivity`.

5.1.5 Perancangan Algoritma

Pada bagian ini menjelaskan tentang penggunaan algoritma yang akan digunakan dalam aplikasi sistem informasi walisongo. Aplikasi ini menggunakan algoritma *Ant Colony Optimization* yang berguna untuk mencari rute tercepat dan algoritma brute force berguna untuk menentukan rute mana yang tercepat dari semua rute yang ditemukan oleh algoritma *Ant Colony Optimization.*

1. Algoritma *Ant Colony Optimization*

Algoritma ini bertindak sebagai pencari rute makam walisongo dengan berdasarkan jumlah siklus dan jumlah semut yang digunakan. Algoritma ini juga berfungsi untuk mencari jarak terdekat dari semua kemungkinan rute yang ada berdasarkan setiap rute yang akan dilalui oleh semua semut yang berhasil menuju lokasi tujuan dari asalnya dan akan menghasilkan rute yang paling pendek dari semua siklus. Selanjutnya akan dijelaskan mengenai perhitungan manual menemukan jalur terpendek.

Proses pertama kali dilakukan adalah melacak posisi koordinat dari *smartphone* pengguna. Nantinya koordinat yang didapatkan kemudian akan disimpan bersamaan dengan koordinat para makam walisongo. Sebagai contoh dapat dilihat pada tabel 5.1

Tabel 5.1 Koordinat awal dan Makam Walisongo

ID	Nama Tempat	Latitude	Longitude	Alamat
1	SMA Semen Gresik	-7.1784971	112.6480804	Gresik
2	Sunan Malik Ibrahim	-7.160205	112.655856	Gresik
3	Sunan Ampel	-7.229882	112.742336	Surabaya
4	Sunan Giri	-7.1697613	112.6307302	Gresik

Waypoint (koordinat) berfungsi untuk menunjukkan lokasi dari suatu tempat lalu dihitung menggunakan *latitude* (lintang) dan *longitude* (bujur). Selanjutnya akan dihitung jarak antara dua *waypoint* seperti berikut :

Mengubah bujur dan lintang dua *waypoint* dari derajat ke radian lalu dibagi dengan 180 / π, atau 57,29578 yang ditunjukkan pada tabel 5.2

Contoh perhitungan :

- SMA Semen Gresik (1)

  Lintang : -7.1784971 / 57.295578 = - 0.1252888503891

  Bujur : 112.6480804 / 57.295578 = 1.9660868138899

Tabel 5.2 Nilai Latitude Radian dan Longitude Radian

ID	Nama Tempat	Lat Radian	Long Radian	Alamat
1	SMA Semen Gresik	-0.1252889	1.966086814	Gresik
2	Sunan Malik Ibrahim	-0.1249696	1.966222524	Gresik
3	Sunan Ampel	-0.1261857	1.96773189	Surabaya
4	Sunan Giri	-0.1251364	1.965783995	Gresik

Setelah diketahui nilai *latitude* dan *longitude* radian maka selanjutnya adalah menghitung jarak sebenarnya dari dua *waypoint* yang ditunjukkan pada tabel 5.3

Contoh Perhitungan :

Hitung jarak dua *waypoint* SMA Semen Gresik dan Sunan Malik Ibrahim

D = Acos (sin (α1) * sin (α2) + cos (α1) * cos (2) * cos (β1− β2)) * R

D = Acos (sin (-0.1252888503891) * sin (-0.1249691513057) + cos (-0.1252888503891) * cos (-0.1249691513057) * cos (1.9660868138899 - 1.9662155921431)) *6371

D = 2.1958307025903

Tabel 5.3 Jarak Antar Kota

kota ke (ID)	1	2	3	4
1	0	2.195	11.937	2.161
2	2.195	0	12.385	2.948
3	11.937	12.385	0	14.096
4	2.161	2.948	14.096	0

Selanjutnya adalah langkah-langkah perhitungan rute terpendek dengan menggunakan Algoritma Ant Colony Optimization.

Parameter yang digunakan adalah :

Alfa (α) = 1

Beta (β) = 2

Rho (ρ) = 0.5

τij awal = 0.01

Maksimum siklus (NCmax) = 1

Tetapan siklus semut (Q) = 1

Banyak semut (m) = 3

Dari jarak kota yang telah diketahui nantinya dapat dihitung visibilitas antar kota (ηij) = 1/dij

Tabel 5.4 Visibilitas Antar Kota

kota ke	1	2	3	4
1	0	0.455581	0.083773	0.462749
2	0.455581	0	0.080743	0.339213
3	0.083773	0.080743	0	0.070942
4	0.462749	0.339213	0.070942	0

Intensitas feromon awal antar titik (τij) ditunjukkan pada tabel 5.5 dibawah ini :

Tabel 5.5 Feromon Awal Antar Titik

kota ke	1	2	3	4
1	0	0.01	0.01	0.01
2	0.01	0	0.01	0.01
3	0.01	0.01	0	0.01
4	0.01	0.01	0.01	0

Nilai dari visibilitas (η) dan intensitas feromon (τ) ini nantinya akan digunakan untuk persamaan probabilitas dan merupakan parameter yang dapat mempengaruhi semut dalam pemilihan titik berikutnya (aturan transisi).

Sebelum sistem melakukan pemetaan kunjungan semut ke setiap titik (kota) hal yang perlu dilakukan adalah menentukan probalitas dari titik 1 ke setiap titik berikutnya dengan persamaan :

$$p_k(r,s) = \left\{ \frac{[\tau(r,s)]^{\alpha}.[\eta(r,s)]^{\beta}}{\sum_{u \epsilon jk(r)}[\tau(r,u)]^{\alpha}.[\eta(r,u)]^{\beta}} \right\} \text{ Jika s } \epsilon \text{ } J_k\,(r) \quad (2.1)$$

Dimana

$$\sum [\tau_{(r,u)}]^{\alpha}.[\eta_{(r,u)}]^{\beta} = ((\tau ij)^{\alpha} * (vkota1)^{\beta}) + ((\tau ij)^{\alpha} * (vkota2)^{\beta}) + ((\tau ij)^{\alpha} * (vkota3)^{\beta}) + ((\tau ij)^{\alpha} * (vkota4)^{\beta})$$

Sehingga hasil perhitungan Σ tiap kota adalah seperti pada tabel 5.6 berikut :

Contoh Perhitungan :

$$\sum [\tau_{(r,u)}]^{\alpha}.[\eta_{(r,u)}]^{\beta} = ((0.01 \text{ ^ } 1) * (0 \text{ ^ } 2)) + ((0.01 \text{ ^ } 1) * (0.455581 \text{ ^ } 2)) + ((0.01 \text{ ^ } 1) * (0.083773 \text{ ^ } 2)) + ((0.01 \text{ ^ } 1) * (0.462749 \text{ ^ } 2))$$

$$\sum [\tau_{(r,u)}]^{\alpha}.[\eta_{(r,u)}]^{\beta} = 0.004287$$

Tabel 5.6 Hasil Perhitungan Σ

Kota	Hasil Σ
1	0.004287
2	0.003291
3	0.000186
4	0.003342

Dengan demikian maka dapat dihitung probabilitas dari kota 1 menuju setiap kota dengan persamaan

Contoh perhitungan dengan menggunakan persamaan 2.1

$$p_k(r,s) = \left\{ \frac{[\tau(r,s)]^{\alpha}.[\eta(r,s)]^{\beta}}{\sum_{u \epsilon jk(r)}[\tau(r,u)]^{\alpha}.[\eta(r,u)]^{\beta}} \right\}$$

$$p_k(1,2) = \left\{ \frac{(0.01)\text{^}1 * (0.455581)\text{^}2}{0.004287} \right\}$$

$$p_k(1,2) = 0.484137932$$

Tabel 5.7 Perhitungan Probabilitas Semut Menuju Kota Selanjutnya

Kota	1	2	3	4
1	0	0.484137932	0.016369966	0.499492102
2	0.630597	0	0.019807465	0.349595594
3	0.377916	0.351069488	0	0.271014998
4	0.640677	0.344265559	0.015057635	0

Setelah hasil perhitungan probabilitas semut menuju kota selanjutnya, maka akan dicari probabilitas kumulutatif (qk), dimana q1 = p1, sedangkan qk = qk-1 + pk, dimana k = 2, 3, ..., n. Kemudian dibangkitkan bilangan random (r) antara 0 sampai 1. Nantinya kota ke-k akan terpilih jika r > qk-1 dan r ≤ qk.

Contoh :

Probabalitas Komulatif dari kota 1 = 0.000 0.484 0.5 1

Bilangan Random (0 – 1) = 0.742

Memeriksa qk-1 < r ≤ qk =

Kota 1 : 0.742 < 0.000 (FALSE)

Kota 2 : 0.742 < 0.484 (FALSE)

Kota 3 : 0.742 < 0.5 (FALSE)

Kota 4 : 0.742 < 1 (TRUE)

Kota yang dipilih adalah kota 4

Setelah semut menyelesaikan sebuah perjalanan, maka dilakukan pembaruan feromon lokal dengan menggunakan persamaan (2,4)

$$\tau(t,v) \leftarrow (1-\rho) * \tau(t,v) + \Delta\tau(t,v)$$

$$\Delta\tau(t,v) = \frac{1}{L_{nn} * c}$$

Dimana :

$\tau(t,v)$ = Jumlah feromon pada sisi dari simpul t ke simpul v

ρ = Tetapan penguapan feromon

$\Delta\tau$ = Perubahan intensitas feromon

L_{nn} = Jarak simpul t ke simpul v

C = Jumlah keseluruhan kota yang dikunjungi

Contoh perhitungan feromon lokal dengan menggunakan persamaan (2,4)

$$\tau(t,v) \leftarrow (1-\rho) * \tau(t,v) + \Delta\tau(t,v)$$

$$\Delta\tau(t,v) = \frac{1}{L_{nn} * c}$$

$$\Delta\tau(1,4) = \frac{1}{2.161 * 4} = 0{,}1138$$

$$\tau(1,4) \leftarrow (1-0.50) * (0.01) + (0.1156)$$

$$\tau(1,4) \leftarrow 0{,}1206$$

Tabel 5.8 Pembaruan Feromon Lokal

Kota	1	2	3	4
1	0	0.1188	0.0259	**0.1206**
2	0.01	0	**0.0251**	0.0898
3	0.01	0.0251	0	0.0227
4	0.01	**0.0898**	0.01	0

Maka hasil rute semut pada siklus 1 seperti pada tabel 5.9

Tabel 5.9 Hasil Pencarian Rute

Semut	Rute	Panjang
1	1,4,2,3	17.494
2	1,2,3,4	28.676
3	1,3,2,4	27.27

Setelah semua semut melakukan perjalan maka tahap selanjutnya adalah melakukan pembaruan feromon global dengan menggunakan persamaan (2,3)

$$\tau(t,v) \leftarrow (1-\alpha).\tau(t,v) + \alpha.\Delta\tau(t,v)$$

$$\Delta\tau(t,v) = \{Lgb^{-1} \;\; jika \; (t,v) \; \epsilon \; rute \; terbaik\}$$

$$\Delta\tau(t,v) = 0$$

Dimana :

$\tau(t,v)$ = Jumlah feromon pada sisi dari simpul t ke simpul v

α = Tetapan pengendali feromon

$\Delta\tau$ = Perubahan intensitas feromon

Lgb = Panjang rute terbaik

Contoh perhitungan feromon global :

Melakukan pembaruan rute terbaik pada siklus 1 :

Rute terbaik adalah **1,4,2,3** dengan panjang *Lgb* = 17.494 km, sehingga

$$\Delta\tau(t,v) = \frac{1}{17.494} = 0.0571$$

$$\tau(t.v) \leftarrow (1-a) * \tau(t,v) + \alpha * \Delta\tau(t,v)$$

$$\tau(1,4) \leftarrow (1-0.1) * (0.1206) + (0.1) * (0.0571)$$

$$\tau(1,4) \leftarrow (0.11425)$$

Melakukan pembaruan feromon global rute lainnya

$$\tau(t.v) \leftarrow (1-a) * \tau(t,v) + \alpha * \Delta\tau(t,v)$$

$$\tau(t.v) = 0$$

$$\tau(1,2) \leftarrow (1-0.1) * (0.1188) + (0.1) * (0)$$

$$\tau(1,2) \leftarrow (0.10692)$$

Tabel 5.10 Pembaruan Feromon Global

Kota	1	2	3	4
1	0	0.10692	0.02331	**0.11425**
2	0.01	0	**0.283**	0.01
3	0.01	0.01	0	0.02043
4	0.01	**0.08653**	0.01	0

Dari tabel diatas dapat diketahui bahwa terjadi perubahan nilai feromon. Titik – titik yang sering dikunjungi oleh semut terjadi penambahan intensitas feromon, sedangkan titik yang jarang dikunjungi oleh semut terjadi pengurangan intensitas feromon. Nilai pembaruan intensitas feromon inilah yang nantinya digunakan untuk melakukan perhitungan pada siklus selanjutnya.

2. Algoritma Brute force

Setelah rute terpendek didapatkan dengan Algoritma *Ant Colony Optimization* maka sistem akan menginisialisasikan daftar titik yang tersimpan pada setiap rute menuju walisongo tersebut. Daftar titik yang tersimpan ditunjukkan pada tabel 5.11 berikut :

Tabel 5.11 Daftar Titik Setiap Rute Walisongo

Semut 1	Semut 2	Semut 3
1	1	1
4	2	3
2	3	2
3	4	4
17.494	28.676	27.27

Selanjutnya sistem akan mencari rute Walisongo yang memiliki jumalh titik yang sama dari rute walisongo yang sudah didapatkan dari proses Algoritma *Ant Colony Optimization*. Pencarian rute walisongo yang ditemukan oleh sistem dengan cara melakukan penelusuran ke setiap rute dan melakukan pencocokan dengan pola rute dan jarak yang ditemukan menggunakan Algoritma *Brute Force* ditunjukan pada tabel 5.12

Tabel 5.12 Hasil Pencarian Rute Walisongo

Semut Ke 1

Rute Algoritma Optimization	1-4-2-3
Pola	1-4-2-3
Jarak	17.494

Semut Ke 2

Rute Algoritma Optimization	1-2-3-4
Pola	1-2-3-4
Jarak	28.676

Semut Ke 3

Rute Algoritma Optimization	1-3-2-4
Pola	1-3-2-4
Jarak	27.27

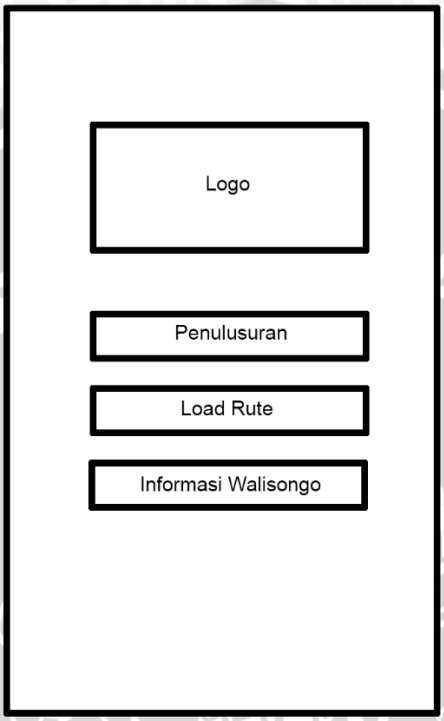
Dari Hasil pencarian maka hasilnya adalah dari SMA Semen Gresik selanjutnya menuju ke Makam Sunan Giri, lalu menuju ke Makam Sunan Maulana Malik Ibrahim dan yang terakhir menuju ke Makam Sunan Ampel.

5.1.6 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk menjelaskan tampilan yang akan di bangun. Berikut penjelasan dari setiap antar muka aplikasi ini :

1. Antarmuka Menu Utama

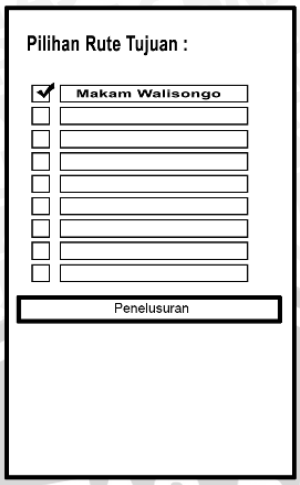
Antarmuka menu utama merupakan sebuah layout yang pertama ditampilkan pada saat aplikasi dijalankan. Pada tampilan ini terdapat tiga menu. Yaitu menu penulusaran, menu load rute dan menu informasi Walisongo. Perancangan antarmuka menu utama ditunjukkan dalam gambar 5.9 berikut :



Gambar 5.9 Antamuka Menu Utama

2. Antarmuka Pilihan Makam Walisongo

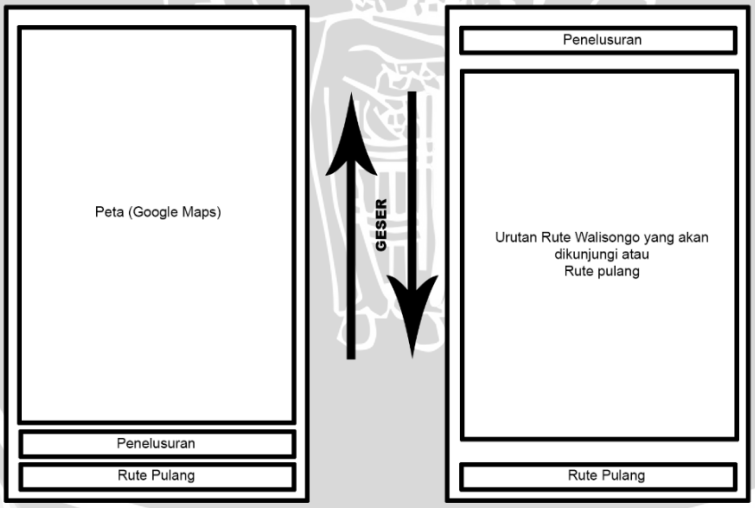
Pada antarmuka ini, aplikasi akan menampilkan menu pilihan Makam Walisongo yang akan dikunjungi. Perancangan tampilan peta ditunjukkan di dalam gambar 5.10



Gambar 5.10 Antarmuka Pilihan Menu Tujuan

3. Antarmuka Peta

Pada antarmuka ini, aplikasi akan menampilkan peta yang berasal dari google maps dengan tipe peta normal. Peta ini secara otomatis akan menampilkan lokasi sekarang dengan bantuan gps. Pada tampilan ini terdapat menu penelusuran dan menu rute pulang. Serta terdapat tombol perbesar dan perkecil peta. Selain itu terdapat garis yang menunjukkan urutan makam walisongo yang akan dikunjungi serta dalam petunjuk di bawah peta dengan menggeser ke atas. Perancangan tampilan peta ditunjukkan di dalam gambar 5.11



Gambar 5.11 Tampilan Peta

4. Antarmuka Load Rute

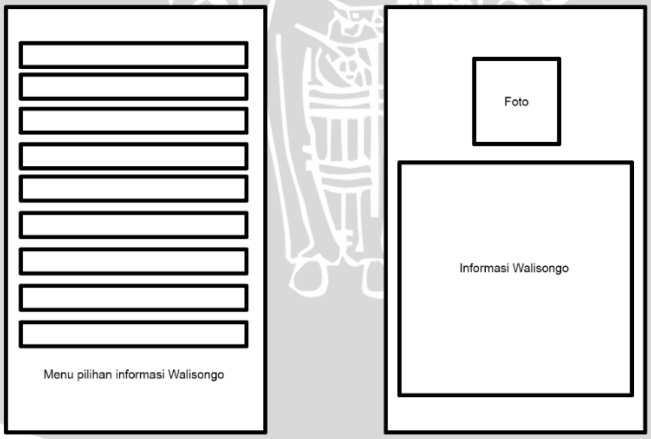
Pada tampilan ini pengguna dapat melihat urutan rute makam walisongo yang akan dikunjungi terlebih dahulu tanpa membuka menu penelusuran. Hal ini dikarenakan menu penelusuran akan mencari ulang lagi dan otomatis titik awal

lokasi akan berbeda. Jadi dengan adanya load rute akan mempermudah pengguna. Selain itu load rute ini disimpan dalam files .txt dalam penyimpanan pada *smartphone*. Tampilan load rute ditunjukkan pada gambar 5.12

Gambar 5.12 Tampilan Load Rute

5. Antarmuka Informasi Walisongo

Pada tampilan ini, pengguna disajikan dengan informasi tentang sejarah walisongo. Diharapkan pengguna dapat mengerti tentang apa itu walisongo juga sejarah penyebaran islam walisongo. Selain informasi, terdapat juga foto para wali. Pada tampilan ini ditunjukkan pada gambar 5.13



Gambar 5.13 Tampilan Informasi Walisongo

5.2 Implementasi

Bagian ini membahas mengenai implementasi sistem berdasarkan hasil yang didapatkan dari analisis kebutuhan dan proses perancangan aplikasi.

Implementasi terdiri atas penjelasan tentang spesifikasi perangkat keras, spesifikasi perangkat lunak dan implementasi fungsi-fungsi utama yang terdiri dari implementasi algoritma dan implementasi antar muka

5.2.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras dipakai dalam proses pembuatan aplikasi dijelaskan dalam tabel 5.13.

Tabel 5.13 Spesifikasi Perangkat Keras

***Notebook* Asus K55VM**	
Processor	Inte Core i5 3210M 2.50 Ghz 64 Bit
Memory (RAM)	4 GB
Harddisk	1 TB
Graphics Processor Unit	NVIDIA GeForce GT 630M 2 GB
***Smartphone* Sony Xperia Z Ultra**	
Chipset	Qualcomm MSM8974 Snapdragon 800
CPU	Quad-core 2.2 GHz Krait 400
GPU	Adreno 330
Memory Internal	16 GB
RAM	2 GB
Display	6,4”
Resolution	1080 x 1920 pixels
GPS	Ada
Jaringan	GSM/HSPA/WiFi

5.2.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pembuatan aplikasi Informasi Walisongo dijelaskan dalam tabel 5.14.

Tabel 5.14 Spesifikasi Perangkat Lunak

Notebook Asus K55 VM	
Operating System	Windows 10 Pro 64 bit
Bahasa pemrograman	Java, HTML, Javascript
Aplikasi yang digunakan	adt bundlewindows-x86 64-20130729 (Eclipse), JDK (Java Development Kit) 7, Adobe Photoshop CC, Notepad ++
Smartphone Sony Xperia Z Ultra	
Operating System	Android
Versi	5.1.1 Lollipop (CM 12.1)

5.2.3 Implementasi Algoritma

Rancang Bangun Aplikasi Informasi Walisongo dibangun dengan menggunakan algoritma *Ant Colony Optimization* dan algoritma *Brute Force* untuk mencari rute tercepat. Selanjutnya dijelaskan bagaimana proses implementasi algoritma *Ant Colony Optimization* dan *algoritma Brute Force*.

5.2.3.1 Algoritma Ant Colony Optimization

Pada tahap ini aplikasi akan menentukan rute walisongo dengan melalui proses penyebaran sejumlah semut yang sudah ditentukan dengan diawali dari titik asal menuju titik tujuan dengan iterasi yang sudah ditentukan. Setiap semut yang melakukan perjalanan dari titik asal akan menentukan rute titik selanjutnya dengan berdasarkan probabilitas titik selanjutnya. Setelah semut sudah melewati semua titik kemungkinan dan sampai ke titik tujuan maka selanjutnya akan dihitung jarak terpendek yang didapat dari semua semut tersebut. Selanjutnya ketika didapatkan jarak terpendek maka akan mengupdate nilai feromone yang berdasarkan titik yang dilewati oleh semut yang mencapai rute terpendek. Setelah semua iterasi dilewati maka akan didapatkan total jarak terpendek dari setiap semut.

Penjelasan algoritma *Ant Colony Optimization*

1. Inisialisasi parameter yang akan digunakan

```
var alfa = 0.1; //tetapan pengendali intensitas jejak semut
var beta = 2.0; //tetapan pengendali visibilitas
var rho = 0.50; //tingkat penguapan jalan feromon
var asymptoteFactor = 0.9; //q0 parameter pembanding bilangan random
var pher = new Array();
var nextPher = new Array();
var prob = new Array();
var numAnts = 10; //jumlah semut
var numWaves = 10; // jumlah siklus
```

Gambar 5.14 Parameter Algoritma *Ant Colony Optimization*

Pada Gambar 5.14 menunjukkan parameter parameter yang digunakan dalam algoritma *Ant Colony Optimization* dimana :

α = 0.1 (Parameter pengendali intensitas jejak semut / feromon)

β = 2.0 (Parameter Pengendali Visibilitas)

ρ = 0.50 (Parameter penguapan feromon global /lokal)

q0 = 0.9 (Parameter pembanding bilangan random)

numAnts = 10 (jumlah semut)

numWaves = 10 (jumlah siklus)

2. Proses perhitungan probabilitas

```
//perhitungan ACO sesuai jumlah semut dan siklus
for (var wave = 0; wave < numWaves; ++wave)
{
for (var ant = 0; ant < numAnts; ++ant)
{
var curr = startNode;
var currDist = 0;
for (var i = 0; i < numActive; ++i)
{
visited[i] = false;
}
currPath[0] = curr;
```

```
for (var step = 0; step < numSteps; ++step)

{

visited[curr] = true;

var cumProb = 0.0;

//Probabilitas

for (var next = 1; next < numValidDests; ++next)

{

if (!visited[next])

{

prob[next] = Math.pow(pher[curr][next], alfa) * Math.pow(dur[curr][next],
0.0 - beta);

cumProb += prob[next];

}

}
```

Gambar 5.15 Perhitungan probabilitas

Tahap selanjutnya adalah mencari probabilitas semut. Variabel `prob[next]` merupakan perhitungan probabilatas yand dilakukan sesuai dengan jumlah semut dan siklus. Nantinya akan di hitung dari tiap semut yang melakukan perjalanan. Selanjutnya dari hasil probabiltas dari semut maka akan dicari probabilitas komulatifnya dengan variabel `cumProp` hingga semua semut dan siklus selesai melakukan perjalanan.

3. Langkah selanjutnya adalah mengambil bilangan random untuk menentukan kota yang menjadi tujuan

```
var guess = Math.random() * cumProb;

var nextI = -1;

for (var next = 1; next < numValidDests; ++next)

{

if (!visited[next])

{

nextI = next;

guess -= prob[next];

if (guess < 0)

{

nextI = next;

break; }
```

Gambar 5.16 Mengambil Bilangan Random

Variabel `var guess` melakukan proses pengambilan bilangan random yang akan menentukan kota tujuan selanjutnya. Nantinya akan dilakukan perulangan untuk mengetahui titik yang tersedia. Jika belum maka dilakukan kunjungan terhadap titik tersebut hingga semua titik sudah dilewati.

4. Selanjutnya tahap K2-Opting

```
var lastStep = numActive;
if (mode == 1)
{
lastStep = numActive - 1;
}
var changed = true;
var i = 0;
while (changed)
{
changed = false;
for (; i < lastStep - 2 && !changed; ++i)
{
var cost = dur[currPath[i+1]][currPath[i+2]];
var revCost = dur[currPath[i+2]][currPath[i+1]];
var iCost = dur[currPath[i]][currPath[i+1]];
var tmp, nowCost, newCost;
for (var j = i+2; j < lastStep && !changed; ++j)
{
nowCost = cost + iCost + dur[currPath[j]][currPath[j+1]];
newCost = revCost + dur[currPath[i]][currPath[j]] + dur[currPath[i+1]][currPath[j+1]];
if (nowCost > newCost)
{
currDist += newCost - nowCost;
for (var k = 0; k < Math.floor((j-i)/2); ++k)
{
tmp = currPath[i+1+k];
currPath[i+1+k] = currPath[j-k];
currPath[j-k] = tmp;
}
changed = true;
--i;
```

```
}
cost += dur[currPath[j]][currPath[j+1]];
revCost += dur[currPath[j+1]][currPath[j]];
```

Gambar 5.17 Tahap K2-Opting

K2-Opting digunakan pada saat semut telah menyelasaikan perjalanannya. Nantinya akan dipilih dua sisi titik dan dilakukan pengecekan apakah titik-titik dapat ditukar untuk membuat rute menjadi lebih baik. Pada tahap ini dilakukan perhitungan jarak yang dibutuhkan dalam sekali perjalanan. Perhitungan jarak sekarang menggunakan variabel `nowCost` dan perhitungan jarak baru dengan variabel `newCost`. Jika jarak sekarang lebih besar dari jarak baru maka dilakukan penukaran hingga dua sisi.

5. Selanjutnya adalah tahap perhitungan update lokal dan penguapan feromon

```
if (currDist < bestTrip)
{
bestPath = currPath;
bestTrip = currDist;
}
//PENGUAPAN FEROMON
for (var i = 0; i <= numSteps; ++i)
{
nextPher[currPath[i]][currPath[i+1]] += (bestTrip - asymptoteFactor *
bestTrip) / (numAnts * (currDist - asymptoteFactor * bestTrip));
}
```

Gambar 5.18 Perhitungan update lokal dan penguapan Feromon

Pada tahap ini, setalah semua semut melakukan penelusuran maka dilakukan perhitungan update lokal dan penguapan feromon seperti pada variabel `nextPher`

6. Tahap perhitungan update feromon global

```
for (var i = 0; i < numActive; ++i) {
for (var j = 0; j < numActive; ++j) {
pher[i][j] = pher[i][j] * (1.0 - rho) + rho * nextPher[i][j];
nextPher[i][j] = 0.0; }
}
```

Gambar 5.19 Perhitungan Update Feromon Global

Setelah dilakukan perhitungan update feromon lokal maka selanjutnya adalah menghitung feromon global yang berguna untuk perhitungan siklus selanjutnya. Untuk perhitungan update feromon global terdapat pada variabel `pher[i][j]`

5.2.3.2 Algoritma *Brute Force*

Pada tahap ini sistem akan menentukan jalur terpendek. Awalnya sistem ini akan menginisialisasikan node – node makam walisongo yang terdaftar serta rute jarak terpendek yang didapatkan dari algoritma *ant colony optimization*. Setelah itu maka sistem akan memulai proses pencarian rute walisongo dengan cara mencoba – coba mana yang memiliki jarak paling pendek dengan syarat semua node harus dikunjungi. Prosesnya dengan cara mencocokan node terdepan dari node rute terpendek tersebut lalu setelah diketahui maka sistem akan memulai menghitung berapa yang memiliki node terdepan yang sama dengan node jarak terpendek dan hasilnya ditemukan node yang memiliki rute dengan jarak paling pendek.

Penjelasan algoritma *Brute Force* :

1. Inisialisasi parameter yang akan digunakan

```
function tspBruteForce(mode, currNode, currLen, currStep)
{
var numSteps = numActive;
var lastNode = 0;
var numToVisit = numActive;
if (mode == 1)
{
numSteps = numActive - 1;
lastNode = numActive - 1;
numToVisit = numActive - 1;
```

Gambar 5.20 Parameter Algoritma *Brute Force*

Pada gambar 5.20 menunjukkan parameter-parameter yang digunakan dalam algoritma *Brute Force* dimana :

numSteps : Jumlah siklus

lastNode : Titik terakhir yang dikunjungi

numToVisit : Jumlah titik yang dikunjungi

numActice : Berisi sejumlah lokasi

2. Jika rute yang terbaik

```
// jika rute yang terbaik:
if (currLen + dur[currNode][lastNode] < bestTrip)
{
// jika ini titik terakhir:
if (currStep == numSteps)
{
currLen += dur[currNode][lastNode];
currPath[currStep] = lastNode;
bestTrip = currLen;
for (var i = 0; i <= numSteps; ++i)
{
bestPath[i] = currPath[i];
}
```

Gambar 5.21 Rute terbaik

Pada tahap ini mencari rute terbaik dengan Algoritma Brute Force, jika panjang jarak keseluruhan lebih kecil dari jarak terbaik maka akan dicari penjumlahan dari variabel `currLen` selanjutnya memilih rute terbaik. Jika bukan rute terbaik maka ke tahap selanjutnya

3. Mencoba semua kemungkinan rute

```
} else
{
// mencoba semua kemungkinan rute
for (var i = 1; i < numToVisit; ++i)
{
if (!visited[i])
{
visited[i] = true;
currPath[currStep] = i;
tspBruteForce(mode, i, currLen+dur[currNode][i], currStep+1);
visited[i] = false;
}
}
```

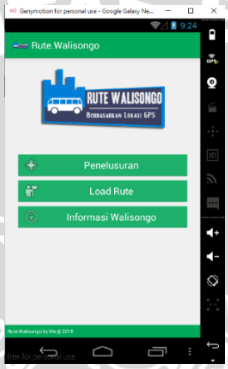
Gambar 5.22 mencoba kemungkinan semua rute

Pada tahap ini akan dilakukan pengecekan semua rute. Tahap pertama adalah memeriksa semua titik, lalu jika belum dikunjungi maka dilakukan perhitungan antar titik hingga titik terakhir pada variabel `tspBruteForce`. Jika tidak memenuhi akan dilakukan pencarian lagi

5.2.4 Implementasi Antarmuka

5.2.4.1 Implementasi Antarmuka Menu

Halaman menu adalah halaman yang pertama kali di kunjungi. Pada halaman menu terdapat pilihan menu seperti penelusuran, load rute, dan informasi walisongo. Antarmuka menu ditunjukkan pada gambar 5.22



Gambar 5.23 Implementasi Antarmuka Menu

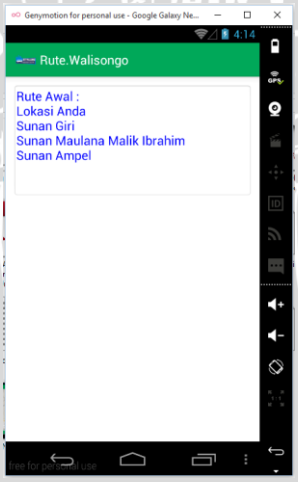
5.2.4.2 Implementasi Antarmuka Menu Penelusuran

Halaman antarmuka menu penelusuran menampilkan peta dan informasi petunjuk arah yang berguna untuk mencari rute tercepat ke Makam para Walisongo dari titik asal pengguna berada. Antarmuka menu penelusuran ditunjukkan pada gambar 5.23

Gambar 5.24 Antarmuka Menu Penelusuran

5.2.4.3 Implementasi Antarmuka Menu Load Rute

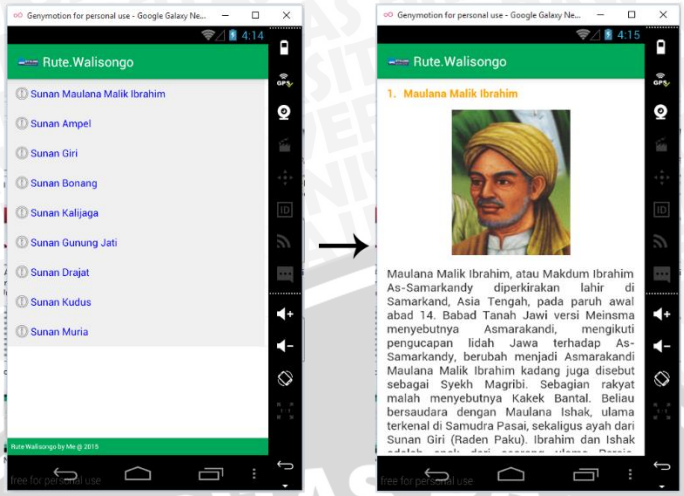
Implementasi Antarmuka *Load Rute* berfungsi untuk menampilkan data penulusuran yang telah di cari terlebih dahulu dan disimpan ke dalam file *Rute.Walisongo.Pergi.txt*. Antarmuka menu *Load Rute* ditunjukkan pada gambar 5.24



Gambar 5.25 Implementasi Antarmuka Menu Load Rute

5.2.4.4 Implementasi Antarmuka Menu Informasi Walisongo

Implementasi antarmuka Menu informasi Walisongo berfungsi untuk menampilkan pilihan menu informasi tentang sejarah singkat para Walisongo. Antarmuka menu informasi Walisongo ditunjukkan pada gambar 5.25



Gambar 5.26 Implementasi Antarmuka Informasi Walisongo

BAB 6 PENGUJIAN DAN ANALISIS

Tahapan terakhir setelah sistem sudah menjadi *prototype* adalah *testing* (pengujian). Pada penelitian ini akan dilakukan lima jenis pengujian yaitu pengujian unit, pengujian integrasi, pengujian validasi, pengujian sistem dan pengujian hardware. Pengujian unit dan integrasi menggunakan metode *whitebox* dengan teknik *basis path testing*. Pada teknik ini proses pengujian dilakukan dengan memodelkan algoritma pada suatu *flow graph* lalu menentukan jumlah *cyclomatic complexity.* Hasil nya untuk menentukan sebuah basis set dan memberikan kasus uji pada setiap basis set yang di tentukan

Pengujian sistem dilakukan dengan melakukan pengujian validasi yaitu dengan menggunakan metode *blackbox*. Pengujian metode *blackbox* adalah metode yang digunakan untuk mengetahui apakah sistem telah berfungsi dengan baik dan sesuai tanpa memperhatikan struktur logika internal perangkat lunak. Sedangkan pengujian validasi berguna untuk mengetahui apakah sistem yang sudah dibangun telah sesuai dengan kebutuhan. Jadi tahapan pengujian validasi dengan menggunakan metode *blackbox* tidak memerlukan konsentrasi khusus terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan kegagalan antara sistem dengan daftar kebutuhan.

Pada pengujian sistem, perangkat lunak yang sudah menjadi *prototype* akan diuji secara mendetail karena perangkat *mobile* sendiri memiliki banyak ukuran layar dan hardware. Bermacam – macam jenis jaringan *mobile* seperti *CDMA* dan *GSM*, memiliki kemampuan jaringan *EDGE,3G,* atau *4G LTE*. Sehingga dilakukan pengujian sistem secara mendetail untuk menemukan apakah sistem sudah berjalan dengan baik.

Lalu yang terakhir adalah pengujian kompatibilitas. Pada pengujian ini akan dilakukan uji coba terhadapa perangkat keras yang terdiri dari versi OS, ukuran layar, resolusi, pengujian terhadap server, serta pengujian aplikasi dalam berinteraksi dengan aplikasi lain pada perangkat keras sehingga dapat memastikan apakah perangkat lunak dapat berjalan dengan baik.

Proses analisis hasil pengujian mengacu pada dasar teori sesuai dengan hasil pengujian yang didapatkan. Analisis dilakukan setelah hasil pengujian diketahui.

6.1 Pengujian

Proses pengujian akan dilakukan melalui lima tahapan, yaitu pengujian unit dan integrasi dengan menggunakan metode *whitebox* dengan teknik *basis path testing* selanjutnya pengujian sistem sacara validasi dengan menggunakan metode *blackbox*, pengujian usability, dan pengujian kompatibilitas. Seluruh proses pengujian ini dilakukan dengan menggunakan spesifikasi perangkat keras dan perangkat lunak sesuai dengan sub bab 3.6 yang sudah dijelaskan sebelumnya.

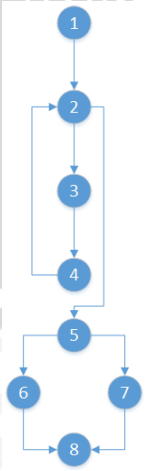
6.1.1 Pengujian Unit

Pada pengujian unit aplikasi sistem informasi Walisongo digunakan teknik *Whitebox testing* dengan teknik *basis path testing*. Pada teknik *basis path testing*, proses pengujian dilakukan dengan memodelkan algoritma pada suatu *flow graph*, menentukan jumlah *cyclomatic complexity*, menentukan sebuah basis set dari jalur independen dan memberikan kasus uji pada setiap basis set yang telah ditemukan. Tabel 6.1 menunjukkan pengujian unit untuk *method* doRute.

Tabel 6.1 Pengujian Unit *Method* doRute

Elemen	Keterangan	Node
Nama	doRute	
Deklarasi	• ArrayList -> listPilihan • CheckBox -> cbTempat	
Prosedur	• Masukkan : CheckBox • Proses 1. ArrayList = List Pilihan 2. For(i = 0, i < cbTempat.leght; i++) a) Jika i adalah cbTempat b) listPilihan.add adalah i 3. EndIf 4. EndFor 5. jika listPilihan.size() kurang dari 3 a) tampilkan “Silahkan pilih lokasi” 6. else a) maka tampilkan kelas MapsActivity 7. EndIf • keluaran : menampilkan kelas MapsActivity	1 2 3 4 5 6 7 8

Gambar 6.1 menunjukkan *flow graph* dari method doRute.



Gambar 6.1 *Flow Grap Method* doRute

Pemodelan ke dalam *flow graph* yang telah dilakukan menghasilkan jumlah *cyclomatic complexity* melalui persamaan

V(G) = E – N +2

Dimana :

V(G) : Jumlah *cyclomatic complexity*

E : Jumlah sisi atau *edge*

N : Jumlah titik atau *node*

Maka perhitungan *cyclomatic complexity* dari *flow graph method* doRute :

V(G) = E – N + 2

V(G) = 9 – 8 + 2

V(G) = 3

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan basis set dari jalur independen, yaitu :

Jalur 1 : 1-2-3-4-2-5-6-8

Jalur 2 : 1-2-3-4-2-5-7-8

Jalur 3 : 1-2-5-7-8

Penentuan kasus uji untuk jalur independen tersebut adalah dan hasil eksekusinya ditunjukkan pada tabel 6.2

Tabel 6.2 Kasus Uji Untuk Pengujian Unit *method* doRute

Jalur	Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan
1	Jika listPilihan 1	Tampilkan “silahkan pilih lokasi”	Tampilkan “silahkan pilih lokasi”
2	Jika listPilihan 4	Tampilkan kelas mapsActivity	Tampilkan kelas mapsActivity
3	Jika listPilihan 5	Melakukan pengecekan apakah terdapat pemilihan opsi tujuan pada list checkbox	Melakukan pengecekan apakah terdapat pemilihan opsi tujuan pada list checkbox dan menampilkan kelas mapsActivity

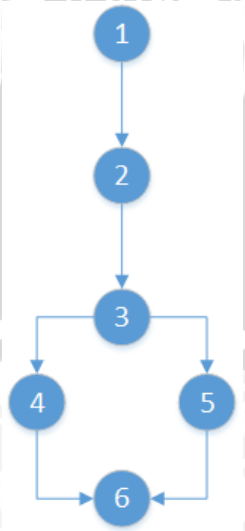
6.1.2 Pengujian Integrasi

Pengujian ini diterapkan pada proses yang mengintegrasikan fungsionalitas dari beberapa *class* untuk melakukan sebuah operasi tertentu. Pada pengujian integrasi yang dijadikan sebagai objek uji adalah *class* inti yang menggabungkan kinerja dari *class* lain. Pengujian integrasi *method* menampilkanRute ditunjukkan pada tabel .

Tabel 6.3 Pengujian Unit *Method* menampilkanRute

Elemen	Keterangan	Node
Nama	menampilkanRute	
Deklarasi	• String -> Lokasi • onPreExecute -> persiapan • Dialog -> dialog • String -> hasil	
Prosedur	• Masukkan : lokasi • Proses 1. Persiapan = dialog.setMessage (“penentuan titik awal”) 1 2. Hasil = hasil.getStartingPoint() 2 3. Jika hasil.toString() adalah kosong 3 a) u.alert(“gagal”) 4 4. Else a) String lokasi = lat + lng b) lokasi = lokasi + hasil.toString 5 c) webview.loadUrl (“file:/android/map.hml”+lokasi) 5. endIf 6. keluaran : gagal atau webview 6	

Gambar menunjukkan *flow graph method* menampilkanRute



Gambar 6.2 *Flow Graph Method* menampilkanRute

Pemodelan ke dalam *flow graph* yang telah dilakukan menghasilkan jumlah *cyclomatic complexity* melalui persamaan

V(G) = E – N +2

Dimana :

V(G) : Jumlah *cyclomatic complexity*

E : Jumlah sisi atau *edge*

N : Jumlah titik atau *node*

Maka perhitungan *cyclomatic complexity* dari *flow graph method* doRute :

V(G) = E – N + 2

V(G) = 6 – 6 + 2

V(G) = 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan basis set dari jalur independen, yaitu :

Jalur 1 : 1-2-3-4-6

Jalur 2 : 1-2-3-5-6

Penentuan kasus uji untuk jalur independen tersebut adalah dan hasil eksekusinya ditunjukkan pada tabel 6.4

Tabel 6.4 Kasus Uji Untuk Pengujian Integrasi *Method* menampilkanRute

Jalur	Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan
1	Jika hasil.toString( ) kosong	Tampilkan “gagal”	Tampilkan “gagal”
2	Jika hasil.toString adalah lokasi	Tampilkan webview file:/android/map.hml dan lokasi	Tampilkan webview file:/android/map.hml dan lokasi

6.1.3 Pengujian Validasi Dengan Metode *Blackbox*

Setelah sistem selesai dibuat maka selanjutnya dilakukan pengujian sistem. Penelitian ini menggunakan pengujian *blackbox* untuk mengetahui fungsi-fungsi khusus yang dirancang untuk mengetahui sebarapa jauh sistem berjalan dan seberapa banyak kesalahan yang ada pada sistem. Pengujian ini berdasarkan dari analisis kebutuhan fungsional sistem pada sub bab 4.4 mengenai daftar spesifikasi persyaratan yang telah dijelaskan sebelumnya. Apabila terjadi kesalahan maka sistem akan segera diperbaiki dan diuji kembali. Hasil dari pengujian validasi ditunjukkan pada Tabel 6.5

Tabel 6.5 Hasil pengujian validasi sistem informasi Walisongo

No	Fungsi	Prasyarat	Skenario yang Dilakukan	Hasil	Status
1	Mencari rute walisongo	*GPS* dan jaringan harus menyala di dalam *smartphone*	• *User* membuka aplikasi • Memilih menu "penelusuran" • Melakukan pemilihan tujuan Makam Walisongo • Melakukan penelusuran rute tercepat	Menampilkan rute tercepat dari titik awal *user* menuju makam para walisongo. Dan juga menampilkan rute pulang serta dapat menyimpan rute tercepat ke dalam file.txt	Valid
2	Melihat rute perjalan	Sudah melakukan pencarian rute walisongo terlebih dahulu	• *User* membuka aplikasi • Memilih menu "load rute"	Menampilkan tujuan yang akan dituju terlebih dahulu dari hasil pencarian rute tercepat yang telah tersimpan ke dalam file.txt	Valid
3	Melihat Informasi tentang walisongo	Tidak ada	• *User* membuka aplikasi • Memilih menu "Informasi Walisongo" • Memilih salah satu pilihan walisongo	Menampilkan informasi tentang walisongo yang berisi foto dan sejarah singkat para walisongo	Valid

6.1.4 Pengujian *Usability*

Pada tahap ini, perangkat lunak yang sudah dibuat akan diuji untuk menilai kepuasan para peziarah dengan hasil rekomendasi yang diberikan oleh system apakah sudah sesuai dengan yang diharapkan dan juga pengujian ini dilakukan untuk menilai seberapa mudah antar muka pengguna dapat digunakan serta untuk menilai hasil pencarian rute pada aplikasi Walisongo. Pengujian ini dilakukan dengan cara membagikan kepada para peziarah sebanyak 30 koresponden. Jenis kuisioner yang digunakan adalah SUS (*System Usability Scale*). Komponen pernyataan yang digunakan untuk pengujian usability dalam kemudaan pengguna ditunjukkan pada tabel 6.6

Tabel 6.6 Komponen Pertanyaan Pengujian *Usability* Kemudahan Penggunaan

NO	DAFTAR PERNYATAAN	ALTERNATIF JAWABAN				
		STS	TS	N	S	SS
		1	2	3	4	5
1	Saya akan menggunakan sistem informasi walisongo untuk mencari rute menuju makam para walisongo					
2	Menurut Saya, sistem informasi walisongo terlalu rumit padahal dapat dibuat lebih sederhana					
3	Sistem Aplikasi informasi walisongo dapat saya gunakan dengan mudah					
4	Saya akan membutuhkan bantuan orang lain saat mengunakan sistem informasi walisongo					
5	Menurut Saya, fungsi-fungsi pada aplikasi walisongo telah terintegrasi dengan baik					
6	Menurut Saya terdapat beberapa fungsi yang tidak sesuai pada sistem aplikasi informasi walisongo					
7	Saya yakin banyak orang yang akan cepat mengerti dan mudah menggunakan sistem aplikasi informasi walisongo					
8	menurut Saya, sistem aplikasi informasi walisongo tidak praktis untuk digunakan					

9	Saya percaya diri dalam mengunakan sistem informasi walisongo					
10	Saya harus mempelajari fungsi dari aplikasi agar terbiasa menggunakan					

Hasil total jawaban dari pernyataan pada tabel 6.6 untuk pengujian *usability* dalam menguji kemudaan dalam penggunaan sistem dengan 30 responden didapatkan hasil total jawaban sebanyak 1022. Perhitungan menggunakan standart penilaian SUS dari hasil pengujian untuk menilai pernyataan pengguna seperti dibawah ini :

Hasil total skor responden = 1022 * 2.5

= 2555

Rata – rata hasil skor SUS = 2555 / 30

= 85.16666667

Jadi hasil pengujian untuk menilai kemudahan penggunaan sistem informasi walisongo didapatkan tingkat kepuasan pengguna sebesar 85.167. Selanjutnya, pengujian *usability* untuk menilai kepuasan hasil sistem pencarian rute walisongo. Komponen pernyataan yang digunakan untuk pengujian *usability* dari sistem pencarian rute walisongo ditunjukkan pada tabel 6.7

Tabel 6.7 Komponen Pertanyaan Pengujian *Usability* Hasil Pencarian Rute

NO	DAFTAR PERNYATAAN	ALTERNATIF JAWABAN				
		STS	TS	N	S	SS
		1	2	3	4	5
1	Apakah hasil pencarian rute yang diberikan oleh sistem terdapat pada posisi anda sekarang					
2	Apakah Anda puas dengan hasil yang diberikan system					
3	Apakah hasil pencarian rute yang dihasilkan sudah sesuai dengan lokasi ziarah walisongo yang anda masukkan					
4	Apakah Anda akan menggunakan hasil pencarian rute pada sistem ini untuk nantinya menentukan arah menuju lokasi ziarah walisongo					

5	Apakah Anda akan memberitahu orang lain akan kepuasan hasil pencarian rute yang diberikan oleh system					

Hasil total jawaban dari pernyataan pada tabel 6.7 untuk pengujian *usability* dalam menguji hasil dari sistem pencarian rute walisongo dengan responden sebanyak 30 didapatkan hasil total 505. Perhitungan menggunakan standart nilai SUS dari hasil pengujian adalah sebagai berikut :

Hasil total skor responden = 536 * 5

= 2680

Rata-rata hasil skor SUS = 2680 / 30

= 89.333

Jadi hasil pengujian untuk menilai hasil sistem pencarian rute tercepat yang dilakukan menghasilkan tingkat kepuasan pengguna sebesar 89.333 dan untuk pertanyaan nomor 2 dan 4 dari komponen pengujian *usability* hasil pencarian rute menunjukkan bahwa nilai untuk kinerja sistem informasi walisongo adalah sebagai berikut :

Hasil total skor responden = 218 * 10

= 2180

Rata-rata hasil skor SUS = 2180 / 30

= 72.6667

Jadi untuk nilai kinerja dari sistem pencarian walisongo menghasilkan tingkat kepuasan pengguna sebesar 72.6667. Nantinya rincian data skor dari jawaban hasil kuisioner akan ditunjukkan pada lampiran.

6.1.3 Pengujian Kompatibilitas

Pengujian kompatibilitas dilakukan karena setiap perangkat keras android memiliki banyak perbedaan spesifikasi seperti versi OS, ukuran layar, resolusi, pengujian terhadap perubahan server dan pengujian aplikasi dalam berinteraksi dengan aplikasi lain pada perangkat keras sehingga dapat memastikan apakah aplikasi dapat berjaan dengan benar. Pada pengujian ini, penulis melakukan uji coba terhadap 10 *smartphone* dan 2 *tablet* sehingga nantinya dapat menaganalisis apakah perangkat lunak sudah sesuai dengan daftar spesifikasi persyaratan atau belum. Berikut daftar-daftarnya :

1. Daftar *Smarphone*

- Sony Ericsson Xperia Arc S

  GSM/HSPA, Wifi, A-GPS, 4.2”, 480 x 854 pixels, Android OS 2.3.4 (Gingerbread), Snapdragon S2, Single core 1,4 Ghz Scorpion, Adreno 205, Internal 1 GB, 512 MB RAM.

- Sony Ericsson Xperia Play

  GSM/HSPA, WiFI, A-GPS, 4.0”, 480 x 854 pixels, Android OS 2.3.4 (Gingerbread), Snapdragon S2, Single core 1 GHz Scorpion, Adreno 205, Internal 400 MB, 512 MB RAM.

- Sony Xperia Z Ultra

  GSM/HSPA (C6802), WiFi, A-GPS, 6.4”, 1080 x 1920 pixels, Android OS 5.1.1 (Lollipop custom rom CM 12.1), Snapdragon 800, Quad-core 2.2 Ghz krait 400, Adreno 330, Internal 16 GB, 2 GB RAM.

- Sony Xperia Z1

  GSM/HSPA/LTE, WiFi, A-GPS, 5.0”, 1080 x 1920 pixels, Android OS 5.1 (Lollipop), Snapdragon 800, Quad-Core 2.2 GHz Krait 400, Adreno 330, Internal 16 GB, 2 GB RAM.

- Lenovo K900

  GSM/HSPA, WiFi, A-GPS, 5.5”, 1080 x 1920 pixels, Android OS 4.2 (Jelly Bean), Intel Atom Z2580, Dual Core 2 GHz, PowerVR SGX544MP2, Internal 16 GB, 2 GB RAM.

- Asus Zenfone 4

  GSM/HSPA, WiFi, A-GPS, 4.0”, 480 x 800 pixels, Android OS 4.4.2 (Kitkat), Intel Atom Z2520, Dual Core 1.2 GHz, PowerVR SGX544MP2, Internal 4 GB, 1 GB RAM.

- Asus Zenfone 5

  GSM/HSPA, WiFi, A-GPS, 5.0”, 720 x 1280 pixels, Android OS 4.4.2 (Kitkat), Intel Atom Z2580, Dual Core 2 GHz, PowerVR SGX544MP2, Internal 16 GB, 2 GB RAM.

- Asus Zenfone Laser 2 (ZE500KL)

  GSM/HSPA/LTE, WiFi, A-GPS, 5.0”, 720 x 1280 pixels, Android OS 5.0.2 (Lollipop), Snapdragon 410, Quad Core 1,2 Ghz, Adreno 306, Internal 16 GB, 2 GB RAM.

- HTC Evo 3D

  GSM/HSPA, WiFi, A-GPS, 4.3”, 560 x 960 pixels, Android OS 4.0 (Ice Cream Sandwich), Snapdragon S3, Dual Core 1.2 Ghz, Adreno 220, Internal 1 GB, 1 GB RAM.

- HTC One M8

  GSM/HSPA/LTE, WiFi, A-GPS, 5.0”, 1080 x 1920 pixels, Android OS 5.0 (Lollipop), Snapdragon 801, Quad Core 2.5 Ghz Krait 400, Adreno 330, Internal 32 GB, 2 GB RAM.

2. Daftar Tablet

- Samsung P7500 Galaxy Tab 10.1 3G

  GSM/HSPA, WiFi, A-GPS, 10”, 800 x 1280 pixels, Android OS 4.0.4 (Ice Cream Sandwich), Nvidia Tegra 2 T20, Dual Core 1 GHz, ULP GeForce, Internal 16 GB, 1 GB RAM.

- Samsung Galaxy Tab 2 7.0 P3100

  GSM/HSPA, WiFi, A-GPS, 7.0”, 600 x 1024 pixels, Android OS 5.1.1 (Lollipop Custom Rom CM 12.1), TI OMAP 4430, Dual Core 1 GHz, PowerVR SGX540, Internal 8 GB, 1 GB RAM.

Lebih Jelasnya di tunjukkan pada tabel 6.8 dan 6.9 :

Tabel 6.8 Pengujian Smartphone

No	Smartphone	Jaringan	GPS	Layar	Resolusi	Prosesor	RAM	Versi OS	Hasil
1	Xperia Arc S	GSM/HSPA, WiFi	Ada	4.2"	480 x 854 px	Snapdragon S2	512 MB	2.3.4	Valid
2	Xperia Play	GSM/HSPA, WiFi	Ada	4.0"	480 x 854 px	Snapdragon S2	512 MB	2.3.4	Valid
3	Xperia Z Ultra	GSM/HSPA, WiFi	Ada	6.4"	1080 x 1920 px	Snapdragon 800	2 GB	5.1.1	Valid
4	Xperia Z1	GSM/HSPA/LTE, WiFi	Ada	5.0"	1080 x 1920 px	Snapdragon 800	2 GB	5.1	Valid
5	Lenovo K900	GSM/HSPA, WiFi	Ada	5.5"	1080 x 1920 px	Intel Atom Z2580	2 GB	4.2	Valid
6	Zenfone 4	GSM/HSPA, WiFi	Ada	4.0"	480 x 800 px	Intel Atom Z2520	1 GB	4.4.2	Valid
7	Zenfone 5	GSM/HSPA, WiFi	Ada	5.0"	720 x 1280 px	Intel Atom Z2580	2 GB	4.4.2	Valid
8	Zenfone Laser 2	GSM/HSPA/LTE, WiFi	Ada	5.0"	720 x 1280 px	Snapdragon 410	2 GB	5.0.2	Valid
9	HTC Evo 3D	GSM/HSPA, WiFi	Ada	4.3"	560 x 960 px	Snapdragon S3	1 GB	4.0	Valid
10	HTC One M8	GSM/HSPA/LTE, WiFi	Ada	5.0"	1080 x 1920 px	Snapdragon 801	2 GB	5.0	Valid

Tabel 6.9 Pengujian Tablet

No	Tablet	Jaringan	GPS	Layar	Resolusi	Prosesor	RAM	Versi OS	Hasil
1	Galaxy Tab 10.1	GSM/HSPA, WiFi	Ada	10.0"	800 x 1280 px	Nvidia Tegra 2 T20	1 GB	4.0.4	Valid
2	Galaxy Tab 2 7.0	GSM/HSPA, WiFi	Ada	7.0"	600 x 1024 px	TI OMAP 4430	1 GB	5.1.1	Valid

Dari hasil pengujian kompatibilitas, dapat dilihat bahwa aplikasi walisongo dapat berjalan dengan lancar di semua perangkat keras. Ini menunjukkan bahwa apa yang ada di dalam spesifikasi persyaratan semuanya terpenuhi dan dapat digunakan dengan baik di perangkat keras android.

6.2 Analisis

Analisis bertujuan untuk menganalisa data hasil pengujian hingga menghasilkan kesimpulan-kesimpulan. Analisis ini dilakukan berdasarkan hasil implementasi dan pengujian yang telah dilakukan terhadap aplikasi dan mengacu pada dasar teori. Proses analisis yang dilakukan adalah analisis hasi pengujian unit dan integrasi (*white box*), analisis hasil pengujian validasi (*black box*), analisis hasil pengujian sistem dan analisis hasil pengujian kompatibilitas.

Berdasarkan hasil pengujian unit dan integrasi menunjukkan bahwa berdasarkan dari kasus uji untuk pengujian unit dan integrasi menunjukkan hasil sesuai dengan yang diharapkan.

Berdasarkan hasil pengujian validasi *sistem*. Implementasi pada perancangan sistem dapat dikatakan layak dan berhasil untuk setiap fungsinya. Sistem dapat menjalankan fungsi-fungsi yang ada dan memproses sesuai dengan spesifikasi persyaratan sistem. Selain itu hasil yang dikeluarkan telah sesuai dan tidak ada kendala sama sekali.

Pada tahap ini, perangkat lunak yang sudah dibuat akan diuji untuk menilai kepuasan para peziarah dengan hasil rekomendasi yang diberikan oleh system apakah sudah sesuai dengan yang diharapkan dan juga pengujian ini dilakukan untuk menilai seberapa mudah antar muka pengguna dapat digunakan serta untuk menilai hasil pencarian rute pada aplikasi Walisongo. Pengujian ini dilakukan dengan cara membagikan kepada para peziarah sebanyak 30 koresponden. Jenis kuisioner yang digunakan adalah SUS (*System Usability Scale*). Penilaian Skor SUS adalah pertama menjumlahkan skor dari setiap item, setiap item memiliki skor berkisar dari 0 sampai 4. Untuk item bernomor ganjil nilai skor adalah posisi skala yang dipilih responden dikurangi 1. Untuk item bernomor genap nilainya 5 dikurangi posisi skala yang dipilih responden. Lalu kalikan jumlah skor sebesar 2.5 untuk mendapatkan nilai keseluruhan SUS. Skor SUS memiliki kisaran 0 hingga 100. Setiap pernyataan memiliki jawaban “Sangat Tidak Setuju”, “Tidak Setuju”, “Netral”, “Setuju”, “Sangat Setuju”.

Dari hasil komponen pertanyaan pengujian *usability* kemudahan pengguna didapatkan tingkat kepuasan pengguna sebesar 85%. Untuk komponen pertanyaan pengujian *usability* hasil sistem pencarian rute walisongo didapatkan tingkat kepuasan pengguna sebesar 89%.

Berdasarkan hasil pengujian kompatibilitas dengan menggunakan 10 *smartphone* dan 2 *tablet* dengan spesifikasi berbeda seperti versi OS, ukuran layar, resolusinya, pengujian terhadap perubahan server, serta pengujian aplikasi dalam berinteraksi dengan aplikasi lain pada perangkat keras. menghasilkan kesimpulan bahwa dari 10 *Smartphone* dan 2 *Tablet* aplikasi sistem informasi Walisongo dapat berjalan dengan baik tanpa kendala.

Secara keseluruhan, sistem dapat dikatakan layak. Hasil pengujian sistem secara keseluruhan menunjukkan bahwa aplikasi dapat bekerja sesuai dengan parameter uji. Semuanya bernilai valid. Dimulai dari pengujian unit dan integrasi (*white* box), pengujian validasi (*Black Box*), pengujian usability, dan pengujian kompatibilitas. Semua dapat berjalan dengan baik. Selain itu, sistem akan bekerja lebih optimal apabila didukung dengan kecepatan akses internet untuk mengunci sinyal GPS dan menampilkannya dalam bentuk *Google Maps.*

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, pengujian dan analisis yang dilakukan, maka dapat diambil kesimpulan sebagai berikut :

1. Dalam mengimplemtasikan *Location Based Service* pada sistem informasi Walisongo berbasis Smartphone Android maka selain fitur informasi mengenai sejarah walisongo, terdapat fitur mencari rute perjalanan terpendek untuk berziarah ke makam para Walisongo dengan menggunakan *Location Based Service* dari *API Google Maps*. Dalam pencarian rute, Algoritma *Ant Colony Optimization* dan Algoritma *Brute Force* diterapkan untuk membantu proses dari *API Google Maps*.
2. Sistem informasi Walisongo mengimplementasikan algoritma Ant Colony Optimization dan algoritma Brute Force. Setelah dilakukan pemilihan tujuan oleh user selanjutnya aplikasi akan menampilkan halaman web di dalam aplikasi untuk melakukan proses pencarian. Pada proses ini algoritma Ant Colony Optimization bekerja untuk mencari rute terpendek dengan variabel 10 semut (numAnts) dan 10 siklus perjalanan semut (numWaves) untuk menghasil beberapa pilihan rute, termasuk rute terpendek. Selanjutnya, Algoritma Brute Force menyeleksi rute-rute tersebut secara langsung dan menghasilkan rute terpendek yang nantinya akan membantu pengguna menuju ke tujuan, yaitu makam para Walisongo.
3. Hasil pengujian dapat disimpulkan bahwa unit modul dan integrasi dari beberapa unit modul dari sistem sudah memenuhi kebutuhan fungsional dengan menghitung kompleksitas siklomatis, selanjutnya pada pengujian validasi menggunakan metode *black box* pada sistem informasi Walisongo telah valid. Berdasarkan pengujian *usability* dengan memberikan kuesioner *system usability scale* (SUS) kepada 30 responden dan menunjukkan bahwa pengujian kemudahan penggunaan aplikasi didapatkan presntasi nilai sebesar 85% dan untuk pengujian hasil pencarian rute didapatkan presentasi nilai 89%. Pada pengujian kompatibilitas menunjukkan bahwa aplikasi dapat berjalan di semua perangkat android dengan OS Android 2.3 keatas.

7.2 Saran

Saran yang dapat diberikan setelah menyelesaikan penelitian skripsi ini nantinya aplikasi akan terus diperbaiki sehingga lebih banyak fitur yang diberikan dan dapat diterapkan pada destinasi pariwisata lainnya.

Daftar Pustaka

Adi, S, & Anwar, H, 2011. *Tradisi Ziarah Makam Walisongo*. KOMPAS,1 Agustus 2011.< http://oase.kompas.com/read/2011/08/01/07003591/Tradisi.Ziarah.Makam.Walisongo.> [Diakses 4 Januari 2015]

Brooke, J, 2013. SUS: A Retrospective. *Journal of Usability Studies Vol 8, Issue 2, 29-40,* [online] Tersedia di : < http://uxpajournal.org/sus-a-retrospective/> [Diakses 4 Januari 2015]

Dorigo, M. & Gambardella, L.M. 1997. Ieee Transactions On Evolutionary Computation. *Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem*. [e-journal] Vol 1. tersedia di : < http://code.ulb.ac.be/dbfiles/DorGam1997tec.pdf > [diakses 4 Januari 2015]

Dorigo, M. & Stutzle,T. 2004. *Ant Colony Optimization*. Massachusetts.: The MIT Press

Dorigo, M. & Stutzle,T.2002. Hand Book of Metaheuristic. *The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances.* Pg 251-258. Dordrecht, Netherland : Kluwer Academic Publishers

Dorigo, M., Maniezzo, V., dan Colorni, A. (1991). Technical report 91-016. *The Ant System: An autocatalytic optimizing process*. Tersedia di < http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.51.4214 > [diakses 4 Januari 2015]

El-Rabbany, A, 2006. *Introduction To GPS : The Global Positioning System, Second Edition*. Norwood: Artech House.

Gatsou, C., Politis, A,. & Zevgolis, D,. 2013. *Exploring Inexperienced User Performance Of A Mobile Tablet Application Through Usability Testing*. In : FedCSIS (Federated Conference on Computer Science And Information System) , 2013. *Proceedings of the 2013 Federated Conference on Computer Science And Information System*. Kraków, Poland, 8-11 September 2013. USA: IEEE Xplore

Gebweb, 2007. *Behind The Scenes Of Optimap*. [online] tersedia di : < http://gebweb.net/blogpost/2007/07/05/behind-the-scenes-of-optimap/ > [diakses 20 Januari 2015]

Irwansyah, C, 2014. *Pencarian Rute Angkutan Umum Menggunakan Algoritma Ant Colony Optimization*. S1. Universitas Brawijaya Malang.

Kumar, M. & Chauhan, M. 2013. *Best Practice in Mobile Application Testing.* Bangalore, India : Infosys. Tersedia di : < http://assets.libertyellisfoundation.org/cms/YETMO7P9-SDCI-0MH3231TPVEA.pdf > [diakses 4 Januari 2015]

Leksono, A, 2009. *Algoritma Ant Colony Optimization (ACO) Untuk Menyelesaikan Traveling Salesman Problem (TSP)*.S1. Universitas Diponegoro Semarang

Munir, R, 2010. *Matematika Diskrit (Revisi Keempat)*. Bandung : Informatika

Prahasta, E, 2005. *Konsep – Konsep Dasar Sistem Informasi Geografis*. Bandung: Informatika.

Prayitno, W, 2005. *Desain Model Sistem Perangkat Lunak Dengan UML*. Departemen Teknik Informatika, Institut teknologi Bandung

Pressman, R.S., 2010. *Software Engineering : A Practioner's Approach, Seventh Edition*. NewYork: McGraw Hill

Riyanto, M, 2010. *Membuat Sendiri Mobile GIS Platform Java ME, Blackberry, Dan Android*. Yogyakarta: CV ANDI Offset.

Safaat, N.H., 2013. *Aplikasi Berbasis Android : Berbagai Implementasi Dan Pengembangan Aplikasi Mobile Android.* Bandung: Informatika.

Simarmata, J, 2010. *Rekayasa Perangkat Lunak*. Yogyakarta: CV Andi Offset

Sirenden, B.H. & Dachi, E.L. 2012. *Buat Sendiri Aplikasi Petamu Menggunakan Codeigniter Dan Google Maps API*. Yogyakarta: CV ANDI Offset.

Siswoyo, B & Andrianto, 2009. Jurnal Computech dan Bisnis, *Studi Komparatif Algoritma Ant Dan Algoritma Genetik Pada Travelling Salesman Problem*. [e-journal] Vol 3 . Tersedia di : < http://jurnal.stmik-mi.ac.id/index.php/jcb/article/view/32/32 > [Diakses 21 Juli 2016]

Stutzle, T., Lopez-Ibanez, M., Pellegrini, P., Maur, M., Montes De Oca, M., Birattari, M., & Dorigo, M. 2010. IRIDIA – Technical Report Series : TR/IRIDIA/2010-002. *Parameter Adaptation In Ant Colony Optimization*. , Belgium : Universite Libre de Bruxelles. Tersedia di : < http://iridia.ulb.ac.be/IridiaTrSeries/rev/IridiaTr2010-002r001.pdf > [diakses 4 Januari 2015]

Sugiyono, 2008. *Metode Penelitian Kuantitatif Kualitatif Dan R&D.* Bandung: Alfabeta

Wahana Komputer, 2013. *Step By Step Menjadi Programmer Android*. Yogyakarta: CV Andi Offset

Witarto, 2004. *Memahami Sistem Informasi*. Bandung: Informatika

LAMPIRAN

Lampiran 1

Rincian Data Skor SUS

Hasil Pengujian *Usability* Kemudahan Penggunaan												Hasil Pengujian *Usability* Hasil Pencarian					
Responden	Pernyataan										Skor	Pernyataan					Skor
	1	2	3	4	5	6	7	8	9	10		1	2	3	4	5	
1	4	3	4	4	4	3	4	3	3	4	36	4	3	3	4	4	18
2	4	4	4	3	3	3	4	3	2	3	33	3	4	3	3	4	17
3	4	4	4	3	4	3	3	2	4	2	33	4	4	3	3	4	18
4	3	4	3	3	3	4	4	3	4	4	35	4	5	4	5	4	22
5	4	3	3	4	3	4	4	4	4	3	36	4	3	4	4	3	18
6	4	3	4	4	4	3	3	2	2	3	32	4	4	2	4	3	17
7	6	2	4	2	3	3	4	3	3	4	34	4	3	2	3	4	16
8	3	2	3	3	3	2	4	4	3	4	31	4	3	3	4	3	17
9	3	2	4	3	4	3	3	3	4	3	32	3	4	4	4	4	19
10	4	3	4	4	3	4	3	4	4	4	37	3	3	3	3	3	15
11	4	3	4	3	4	4	3	3	3	4	35	4	4	4	4	4	20
12	4	2	4	2	4	3	4	2	4	3	32	3	3	3	4	4	17
13	3	4	4	3	4	3	3	3	4	4	35	4	3	3	4	3	17
14	4	3	4	3	3	2	4	4	4	4	35	4	5	3	3	3	18
15	4	3	3	2	4	4	3	4	4	3	34	4	3	4	4	3	18
16	4	3	3	4	3	4	4	3	4	4	36	4	4	4	4	4	20
17	3	4	4	4	4	4	2	3	4	4	36	4	4	4	3	4	19
18	4	4	4	3	3	3	4	2	2	4	33	3	3	5	4	4	19
19	4	3	4	4	4	2	3	2	4	3	33	4	3	4	4	4	19
20	4	3	4	2	3	3	2	3	4	4	32	2	4	2	4	3	15
21	3	2	3	3	3	3	3	2	2	3	27	4	3	4	3	4	18
22	4	3	4	3	4	4	4	2	4	4	36	3	4	3	4	4	18
23	4	3	4	4	4	4	4	4	3	3	37	4	3	3	3	4	17
24	3	4	3	3	4	3	3	3	3	3	32	3	4	3	4	3	17
25	4	4	3	2	4	4	4	3	2	4	34	4	4	4	3	4	19
26	4	4	4	3	3	4	4	3	4	4	37	4	4	3	3	4	18
27	4	3	4	2	4	3	3	3	2	3	31	4	3	2	4	4	17
28	3	4	4	3	3	4	3	4	2	4	34	3	3	3	4	3	16
29	4	4	4	4	4	3	4	4	3	4	38	4	4	4	4	4	20
30	4	3	4	4	4	2	4	4	3	4	36	4	4	3	3	3	17
JUMLAH SKOR											1022	JUMLAH SKOR					536
RATA - RATA HASIL SKOR SUS											85.16667	RATA - RATA HASIL SKOR SUS					89.33333

Lampiran 2

ANGKET INSTRUMEN PENELITIAN

Identitas Responden

(responden tidak perlu menulis nama)

1. No. Responden : _____ (diisi oleh peneliti)
2. Jenis Kelamin : Pria/Wanita *) 3. Usia : ____ Tahun
3. Asal (Kota) : _____

A. Kuesioner Pengujian *Usability* Kemudaan Pengguna

PETUNJUK PENGISIAN :

Berdasarkan atas pengalaman Bapak/Ibu, Berilah tanda silang (X) pada bobot nilai aternatif jawaban pada setiap pernyataan.

Pilihan Respon

1. Sangat Tidak Setuju (STS)
2. Tidak Setuju (TS)
3. Netral (N)
4. Setuju (S)
5. Sangat Setuju (SS)

NO	DAFTAR PERNYATAAN	ALTERNATIF JAWABAN				
		STS	TS	N	S	SS
		1	2	3	4	5
1	Saya akan menggunakan sistem informasi walisongo untuk mencari rute menuju makam para walisongo					
2	Menurut Saya, sistem informasi walisongo terlalu rumit padahal dapat dibuat lebih sederhana					
3	Sistem Aplikasi informasi walisongo dapat saya gunakan dengan mudah					
4	Saya akan membutuhkan bantuan orang lain saat mengunakan sistem informasi walisongo					
5	Menurut Saya, fungsi-fungsi pada aplikasi walisongo telah terintegrasi dengan baik					
6	Menurut Saya terdapat beberapa fungsi yang tidak sesuai pada sistem aplikasi informasi walisongo					

7	Saya yakin banyak orang yang akan cepat mengerti dan mudah menggunakan sistem aplikasi informasi walisongo					
8	menurut Saya, sistem aplikasi informasi walisongo tidak praktis untuk digunakan					
9	Saya percaya diri dalam mengunakan sistem informasi walisongo					
10	Saya harus mempelajari fungsi dari aplikasi agar terbiasa menggunakan					

B. Kuesioner Pengujian *Usability* Hasil Pencarian Rute

PETUNJUK PENGISIAN :

Berdasarkan atas pengalaman Bapak/Ibu, Berilah tanda silang (X) pada bobot nilai aternatif jawaban pada setiap pernyataan.

Pilihan Respon

6. Sangat Tidak Setuju (STS)
7. Tidak Setuju (TS)
8. Netral (N)
9. Setuju (S)
10. Sangat Setuju (SS)

NO	DAFTAR PERNYATAAN	ALTERNATIF JAWABAN				
		STS	TS	N	S	SS
		1	2	3	4	5
1	Apakah hasil pencarian rute yang diberikan oleh sistem terdapat pada posisi anda sekarang					
2	Apakah Anda puas dengan hasil yang diberikan sistem					
3	Apakah hasil pencarian rute yang dihasilkan sudah sesuai dengan lokasi ziarah walisongo yang anda masukkan					
4	Apakah Anda akan menggunakan hasil pencarian rute pada sistem ini untuk nantinya menentukan arah menuju lokasi ziarah walisongo					
5	Apakah Anda akan memberitahu orang lain akan kepuasan hasil pencarian rute yang diberikan oleh system					

repository.ub.ac

UNIVERSITAS BRAWIJAYA