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

Mohammad Rozy Ardiansyah[1], Agi Putra Kharisma,S.T.,M.T.[2], Aswin Suharsono,S.T.,M.T.[3]
Program Studi Informatika/Ilmu Komputer
Fakultas Ilmu Komputer Universitas Brawijaya Malang
Jalan Veteran Malang 65145, Indonesia
Email : rozyk54@hotmail.com

ABSTRACT

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*

ABSTRAK

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

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, pengujian *usability* dan kompatibilitas.

2. TINJAUAN PUSTAKA

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 metode 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* (ASRank), *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 Dasar Teori

2.2.1 *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 dipelajari 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 (Munir, 2010)

2.2.2 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.

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 (5.1), yang memberikan probabilitas semut k di kota r memilih untuk pindah ke kota s. (Dorigo & Stutzle, 2004)

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

Jika s ϵ J_k (r) (5.1)

Dimana :

Pk (r,s) : Probabilitas semut k memilih untuk berpindah dari kota r ke kota s

τ (r,s) : Jumlah feromon pada sisi dari simpul r ke simpul s

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

τ (r,u) : jumlah feromon pada sisi dari simpul r ke simpul u.

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

J_k : Himpunan yang berisi simpul 1– simpul yang telah dikunjungi oleh semut

u : 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 (5.2). (Dorigo & Stutzle, 2004)

$$s = \left\{ \frac{\max\{[\tau(r,u)] \cdot [n(r,u)]^{\beta}\}}{S} \right\} \quad jika\ q \leq q_0 \quad (5.2)$$

Dimana :

τ (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 perbandingan terhadap simpul yang belum ditemuinya

S : simpul berikutnya yang dipilih berdasarkan persamaan (5.1)

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 (5.3). (Dorigo & Stutzle, 2004)

$$\tau(r,s) \leftarrow (1-\alpha) \cdot \tau(r,s) + \alpha \Delta\tau(r,s) \quad (5.3)$$

Dimana :

τ (r,s) : nilai feromon akhir setelah mengalami perubahan

α : tetapan pengendali feromon

Δτ : perubahan intensitas feromon

Pembaruan feromon global ini adalah untuk menyediakan besaran feromon untuk mengunjungi rute terpendek. Persamaan 5.3 menyatakan bahwa hanya sebuah edge rute perjalanan terbaik secara keseluruhan akan mendapat penguatan nilai feromon.

Ketika membuat sebuah solusi TSP, semut akan mengunjungi edge dan mengubah tingkat feromonnya dengan menerapkan aturan memperbarui feromon lokal. Persamaan (5.4) (Dorigo & Stutzle, 2004)

$$\tau(r,s) \leftarrow (1-\rho) \cdot \tau(r,s) + \rho \cdot \Delta\tau(r,s) \quad (5.4)$$

Dimana :

τ (r,s) : jumlah feromon pada sisi dari simpul r ke simpul s

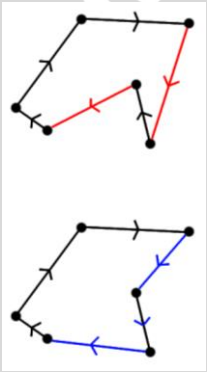
ρ : tetapan penguapan feromon

Δτ : perubahan intensitas feromon

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 menjadi kehilangan intensitas feromon dan jika itu rute tercepat maka intensitas feromonnya 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 1 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).

2.2.3 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.

3. Metodologi

Rancang Bangun Aplikasi Informasi Walisongo berbasis *Location Based Service* pada *Smartphone Android* menggunakan metode penelitian *Research and Development* yang 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.

3.1 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 penelitian ini lebih terurut.

3.2 Algoritma Yang Digunakan

Algoritma yang digunakan adalah Algortima *Ant Colony Optimization* dan Algoritma *Brute Force* yang semuanya berfungsi untuk mencari rute tercepat.

3.3 Subjek Penelitian

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

3.4 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.

3.5 Perangkat Penelitian

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 ++
3. Sistem Operasi
   Windows 10 Pro 64 bit. Versi *Android* yang digunakan adalah 5.0.2 *Lollipop*.

3.6 Metode Pengumpulan Data

Metode pengumpulan data yang digunakan yaitu observasi, studi pustaka, dan kuesioner

3.7 Strategi Dan Analisis Hasil Pengujian

Pada Tahap ini, sistem sudah menjadi prototype dan akan dilakukan pengujian. pengujian pada penelitian ini ditinjau dari pengujian unit, pengujian integritas, pengujian validasi, pengujian *usability*, dan pengujian kompatibilitas.

4. REKAYASA PERSYARATAN

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

4.4 Daftar Spesifikasi Persyaratan

Daftar Spesifikasi Persyaratan dilakukan untuk mendapatkan kebutuhan aplikasi walisongo yang berupa kebutuhan fungsional dan non-fungsional sehingga aplikasi dapat memenuhi kebutuhan pengguna.

Tabel 4.2 Spesifikasi Kebutuhan Fungsional

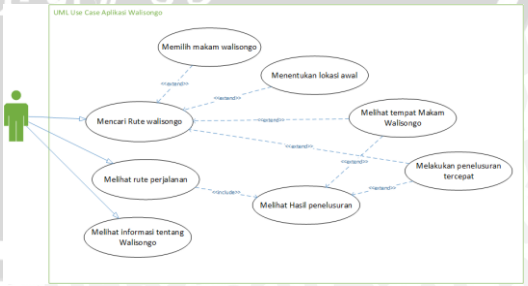
Nomor SRS	Kebutuhan	*Use Case*
SRS_01	Perangkat lunak harus mampu menyediakan fasilitas pencarian rute tercepat dari lokasi awal (user)	Mencari Rute Walisongo
SRS_02	Perangkat lunak dapat menampilkan informasi rute yang di dapat tanpa harus melakukan penelusuran lagi	Melihat rute perjalanan
SRS_03	Perangkat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

4.3 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 4.4



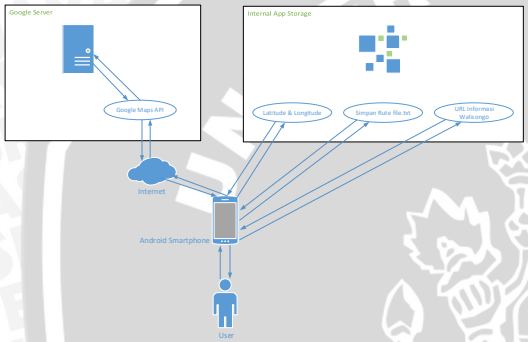
Gambar 4.4 Use Case Diagram

5. PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan Arsitektural

Aplikasi sistem informasi walisongo dengan menggunakan *location based service*

pada sistem operasi *Android* merupakan aplikasi yang dirancang dan dibuat untuk membantu para peziarah dalam mencari rute tercepat ke makam para Walisongo dengan *google maps api* serta menggunakan algoritma *ant colony optimization* dan algoritma *brute force*. Melalui aplikasi tersebut pengguna dapat melakukan pemilihan makam para Walisongo yang akan dikunjungi. Selanjutnya aplikasi akan menentukan titik awal pengguna sebagai pemberangkatan dan melakukan pencarian rute tercepat. Selain mencari rute tercepat, aplikasi juga memberikan informasi mengenai sejarah singkat mengenai para Walisongo. Gambaran umum aplikasi ditunjukkan pada gambar 5.1



Gambar 5.1 Perancangan Arsitektural Sistem

5.2 Implementasi Antarmuka

5.2.1 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 5.2

Gambar 5.2 Implementasi Antarmuka Menu

5.2.2 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 5.3

Gambar 5.3 Implementasi Antarmuka Penelusuran

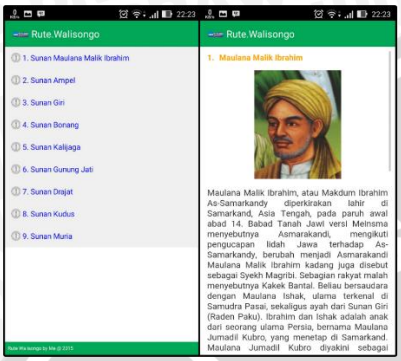
5.2.3 Implementasi Antamuka Menu Load Rute

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 5.4

Gambar 5.4 Implementasi Antarmuka Load Rute

5.2.4 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 5.5



Gambar 5.5 Implementasi Antarmuka Informasi Walisongo

6. PENGUJIAN DAN ANALISIS

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.

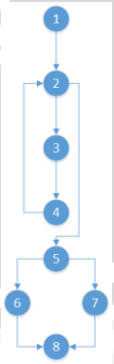
6.1 Pengujian Unit

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--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

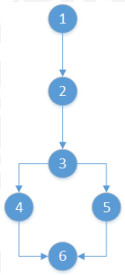
6.2 Pengujian Integrasi

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 6.3

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) 6. keluaran : gagal atau webview	

Gambar 6.2 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

6.3 Pengujian Validasi

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 system sesuai 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	Skenario yang Dilakukan	Status
1	Mencari rute walisongo	• *User* membuka aplikasi • Memilih menu "penelusuran" • Melakukan pemilihan tujuan Makam Walisongo • Melakukan penelusuran rute tercepat	Valid
2	Melihat rute perjalan	• *User* membuka aplikasi • Memilih menu "load rute"	Valid
3	Melihat Informasi tentang walisongo	• *User* membuka aplikasi • Memilih menu "Informasi Walisongo" • Memilih salah satu pilihan walisongo	Valid

6.4 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*). 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%.

6.5 Pengujian Kompatibilitas

Berdasarkan hasil pengujian kompatibilitas dengan menggunakan 10 *smartphone* dan 2 *tablet* dengan spesifikasi berbeda seperti prosesor, memori internal, ukuran layar, resolusinya, besarnya ram, jaringan koneksinya, dll menghasilkan kesimpulan bahwa aplikasi sistem informasi Walisongo dapat berjalan dengan baik tanpa kendala.

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]
- Dorigo, M. & Stutzle,T. 2004. Ant Colony Optimization. Massachusetts.: The MIT Press
- 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, Candra., 2014. Pencarian Rute Angkutan Umum Menggunakan Algoritma Ant Colony Optimization. S1. Universitas Brawijaya.
- 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