

# Pengembangan Aplikasi Optimasi Rute Destinasi Wisata di Banyuwangi Menggunakan *Modern Android Development (MAD) Pattern*

Wahyu Ramadhan<sup>a1</sup>, I Gede Santi Astawa<sup>a2</sup>, Luh Gede Astuti<sup>a3</sup>, Luh Arida Ayu Rahning Putri<sup>a4</sup>, I Putu Gede Hendra Suputra<sup>a5</sup>, I Wayan Santiyasa<sup>a6</sup>

<sup>a</sup>Informatics, Faculty of Math and Science, Udayana University  
South Kuta, Badung Regency, Bali, Indonesia

<sup>1</sup>wahyu.wrseven@gmail.com (*Corresponding author*)

<sup>2</sup>santi.astawa@unud.ac.id

<sup>3</sup>lg.astuti@unud.ac.id

<sup>4</sup>rahningputri@unud.ac.id

<sup>5</sup>hendra.suputra@unud.ac.id

<sup>6</sup>santiyasa@unud.ac.id

## Abstract

*The closest route optimization problem is a classic problem that is still often found in everyday life, one of them is the Traveling Salesman Problem (TSP). This study aims to find solutions in the form of optimal distances to various tourist destinations in Banyuwangi. In this study, the author uses the Python programming language as the backend and the Kotlin programming language as the frontend of the Android application as part of Modern Android Development. The results of the system evaluation, namely Black Box and User Acceptance Test on the application, show that the system that has been carried out can be concluded that the functional system can work well without experiencing bugs or problems on every available feature and page. In designing the application, the author uses the Modern Android Development pattern.*

**Keywords:** TSP, Optimization, Modern, Android, Development

## 1. Pendahuluan

Permasalahan optimasi rute terdekat adalah permasalahan klasik yang sampai saat ini masih sering ditemukan dalam kehidupan sehari-hari, salah satunya adalah *Travelling Salesman Problem (TSP)*. TSP adalah permasalahan umum dalam optimasi kombinatorial dimana seorang *salesman* harus mengunjungi sejumlah N kota, dengan syarat setiap kota hanya dikunjungi sekali. *Salesman* ini harus memilih rute sehingga jarak total yang dia tempuh minimum [1].

Permasalahan *Travelling Salesman* ini dapat diselesaikan dengan menggunakan berbagai algoritme optimasi, salah satunya adalah algoritme *Ant Colony Optimization (ACO)*. Algoritme ini terinspirasi dari pengamatan terhadap suatu koloni semut yang hidup sebagai satu kesatuan koloninya [2]. Solusi untuk TSP telah banyak diterapkan pada kehidupan nyata terutama pada bidang jasa seperti pemilihan rute pendistribusian barang untuk mendapatkan rute yang paling pendek sehingga dapat mengurangi biaya dan waktu transportasi dengan menggunakan berbagai algoritme optimasi.

Penelitian ini dilakukan karena dirasa perlu untuk melakukan penelitian berupa optimasi rute pada destinasi wisata khususnya di Banyuwangi karena saat ini pariwisata di Banyuwangi sudah cukup berkembang sehingga diharapkan hasil pada penelitian ini dapat bermanfaat bagi wisatawan sekaligus dapat menjadi acuan untuk penelitian-penelitian berikutnya. Penggunaan *Modern Android Development (MAD)* sebagai *pattern* dari pengembangan aplikasi dilakukan karena MAD

merupakan *blueprint* untuk para *developer* supaya bisa lebih produktif dan menghasilkan aplikasi yang jauh lebih bagus dan kompatibel untuk jutaan *device* [3].

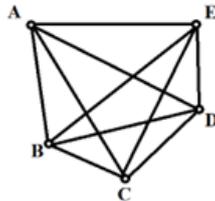
## 2. Metodologi Penelitian

### 2.1. Graf

Teori Graf merupakan suatu diagram yang memuat informasi tertentu jika diinterpretasikan secara tepat. Dalam kehidupan sehari-hari graf digunakan untuk menggambarkan berbagai macam struktur yang ada. Tujuannya adalah sebagai visualisasi objek-objek agar lebih mudah dimengerti. Beberapa contoh graf yang sering dijumpai antara lain struktur organisasi, bagan alir, pengambilan mata kuliah, peta, rangkaian listrik, dan lain-lain [4].

### 2.2. Travelling Salesman Problem

Pada tahun 1800, matematikawan Irlandia William Rowan Hamilton dan matematikawan Inggris Thomas Penyngton mengemukakan permasalahan matematika yang merupakan cikal bakal dari permasalahan *Travelling Salesman Problem* (TSP). Permasalahan TSP kemudian disajikan dalam bentuk permainan yang bernama "Icosian Hamilton" yang mengharuskan pemain untuk menyelesaikan permainan dengan menghubungkan 20 titik perjalanan hanya dengan menggunakan jalur-jalur tertentu. Selanjutnya hal ini dikenal dalam matematika diskrit yang disebut dengan teori Sirkuit Hamilton [5].

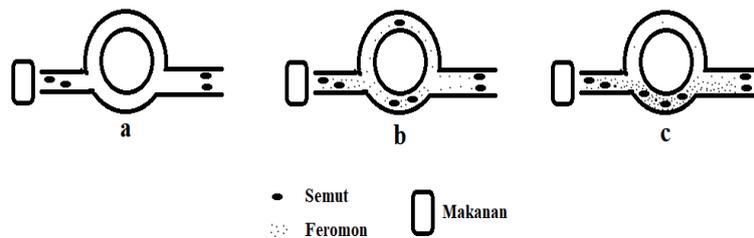


Gambar 1. Contoh Graf TSP

Menurut [6], Persoalan TSP melibatkan seorang *salesman* yang harus melakukan kunjungan pada setiap kota. Rangkaian kota-kota yang dia kunjungi dinamakan lintasan, dimana dalam lintasan tersebut terdapat batasan yaitu tidak boleh ada lebih dari satu kota yang sama. Dengan kata lain, dalam mengunjungi kota-kota, *salesman* tidak boleh singgah pada suatu kota lebih dari satu kali. Misal jika terdapat lima kota yaitu A, B, C, D dan E. Lintasan yang ditempuhnya adalah dari kota A ke kota B ke kota C ke kota D kemudian ke kota E. Penyelesaian dari persoalan ini adalah nilai optimum dari rute yang paling pendek.

### 2.3. Algoritme Ant Colony Optimization

Algoritme *Ant Colony Optimization* (ACO) merupakan algoritme yang dimunculkan sebagai suatu pendekatan multi-agen (*multi-agent approach*) terhadap optimasi berbagai permasalahan yang berkaitan dengan graf. Algoritme ini terinspirasi dari pengamatan terhadap suatu koloni semut yang hidup sebagai satu kesatuan koloninya [2].



Gambar 2. Ilustrasi perjalanan semut dari sarangnya menuju sumber makanan.

### 2.4. Android

Android adalah sistem operasi yang digunakan pada perangkat *mobile* atau selular. Android merupakan platform pemrograman *open-source* yang dikembangkan oleh Google. Sistem operasi Android dapat berjalan di berbagai macam perangkat *mobile* yang berbeda. Android

menyediakan paket pengembangan serta pasar untuk mendistribusikan aplikasi yang dikembangkan. Bahasa pemrograman yang dapat digunakan untuk membuat aplikasi Android antara lain Kotlin, Java, C++, dan lain-lain.

## 2.5. *Modern Android Development*

Perkembangan Android sangat cepat. Apa yang bagus dilakukan 5 tahun lalu, belum tentu masih bagus dilakukan sekarang. Hal itu karena banyak fitur baru yang hadir maupun karena ada *update* dari fitur yang sudah ada. Google telah merilis konsep *Modern Android Development* (MAD). MAD merupakan *blueprint* untuk para *developer* supaya bisa lebih produktif dan menghasilkan aplikasi yang jauh lebih bagus dan kompatibel untuk jutaan *device* [3]. Secara umum, terdapat 4 pilar pada *Modern Android Development* [7], antara lain:

### 2.5.1. *Language*

Dari sisi bahasa, Android telah merekomendasikan Kotlin sebagai bahasa utama setelah resmi menjadi bahasa alternatif. Hal ini merupakan kelebihan Kotlin, yakni seperti bahasanya yang ringkas, cepat, dan aman. Selain itu, karena sifatnya yang *interoperability*, banyak sekali *developer* yang dapat beralih dari bahasa Java ke Kotlin dengan cepat. Hampir 87% dari 1000 aplikasi teratas di Play Store menggunakan Kotlin [7].

### 2.5.2. *Tools*

Dari sisi *tools*, Android Studio menjadi pilihan terbaik untuk mengembangkan aplikasi Android. Hal ini karena ia telah mengakomodasi semua kebutuhan dari awal sampai akhir. Mulai dari membuat *project* dengan *template*, mendesain dengan menggunakan *Layout Editor*, *testing* menggunakan *emulator*, sampai membuat *file* APK/AAB bisa dilakukan di Android Studio.

### 2.5.3. *Distribution*

Secara umum aplikasi Android menggunakan APK sebagai format pendistribusian aplikasi. Namun, format yang terbaik saat ini adalah AAB (Android App Bundle). Sebab ukuran aplikasi yang diunduh bisa menjadi jauh lebih kecil. Hal ini karena sifatnya yang dinamis, di mana ia dapat mengunduh hanya bagian (seperti bahasa, arsitektur, dan *density*) yang diperlukan saja.

### 2.5.4. *APIs*

API pada pengembangan MAD dirangkum dalam *library* Jetpack. Pada Google I/O 2018, Google resmi mempublikasikan Android Jetpack. Jetpack merupakan kumpulan dari komponen *library*, *tools*, dan panduan untuk membuat aplikasi yang bagus

## 3. Hasil dan Pembahasan

### 3.1. Analisis Kebutuhan

Analisis kebutuhan merepresentasikan bagaimana cara sistem bekerja untuk menyelesaikan tugas atau kebutuhan dari pengguna saat menggunakan aplikasi. Kebutuhan ini juga dapat dikatakan sebagai fitur yang ada dalam aplikasi. Dalam pengembangan aplikasi ini terdapat satu fitur utama yaitu melakukan optimasi rute wisata dengan menggunakan algoritme *Ant Colony Optimization*.

#### 3.1.1. Analisis Kebutuhan Fungsional

Kebutuhan fungsional meliputi apa saja fitur-fitur yang harus ada dan merepresentasikan jalannya suatu sistem. Berikut adalah analisis kebutuhan fungsional dari sistem yang dibangun pada penelitian ini.

- a. Sistem harus dapat mengirimkan data *input* berupa *list* lokasi yang akan dilakukan optimasi menggunakan ACO ke server
- b. Server harus dapat mengolah masukan data pengguna dan mengembalikan nilai hasil proses komputasi
- c. Sistem harus dapat menangkap dan menampilkan hasil nilai *return* berupa total jarak dan *list* rute dari server yang kemudian divisualisasikan pada *maps*.

### 3.1.2. Analisis *Kebutuhan Non Fungsional*

Kebutuhan non fungsional berperan untuk membantu fungsionalitas dari sistem itu sendiri. Berikut merupakan analisis kebutuhan non fungsional untuk aplikasi yang dibangun pada penelitian ini.

#### a. *Usability*

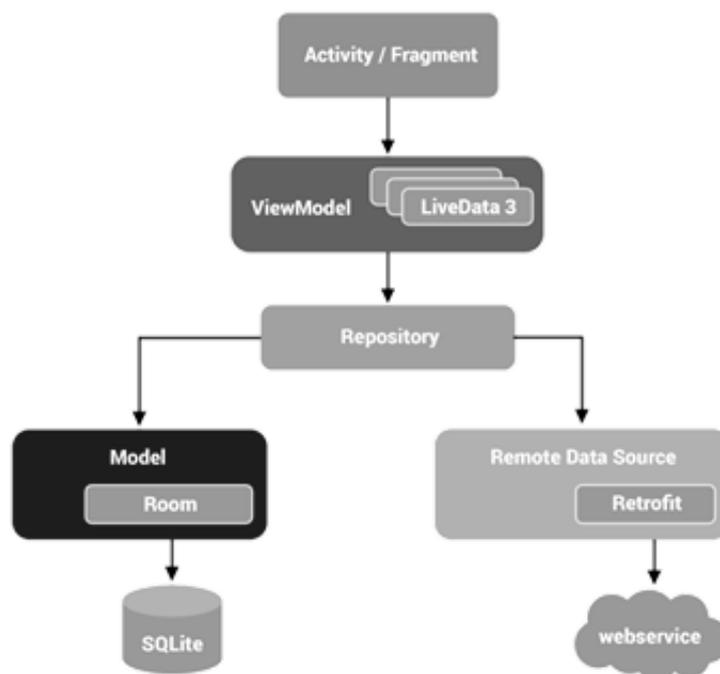
Agar dapat mudah dipahami oleh pengguna, aplikasi harus dilengkapi dengan UI yang simpel, *icon* yang mudah dipahami, serta ukuran teks dan *button* yang dapat dibaca dengan baik.

#### b. *Portability*

Agar sistem dapat berjalan dengan baik pada lintas versi Android *device*, maka sistem yang dibangun harus memerhatikan spesifikasi minimal dari *device* yang dipakai oleh pengguna.

## 3.2. Desain

Pada penelitian ini penulis mengembangkan aplikasi Android dengan menggunakan *architecture component* pada *Modern Android Development*. Adapun desain dari arsitektur aplikasi dapat dilihat pada Gambar 3.



**Gambar 3.** Desain *Architecture Component* MAD

Dapat dilihat pada Gambar 3, terdapat beberapa bagian antaranya *Activity/Fragment*, *Viewmodel* beserta *LiveData*, *Repository*, *Model* beserta *Room* *SQLite* database, *Remote DataSource* beserta *Retrofit* dan *Rest API* yang memiliki tugas sebagai berikut:

1. *Activity* atau *Fragment* bertanggung jawab sebagai *View*
2. *View* akan meminta data serta melakukan observasi terhadap data. Apabila terdapat perubahan terhadap data maka *View* akan bertanggung jawab untuk melakukan *update* pada tampilan sesuai dengan data terbaru.
3. *ViewModel* bertanggung jawab untuk meminta data dan berkomunikasi dengan *Repository* dan menyimpan data dalam bentuk *LiveData* agar *View* dapat melakukan observasi.
4. *Repository* bertanggung jawab sebagai *Model* untuk mendapatkan data atau perubahan data dan melakukan *update* terhadap data yang dimiliki. *Repository* juga bertanggung jawab untuk

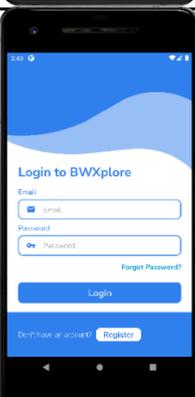
mengatur sumber data yang dibutuhkan oleh aplikasi seperti data yang berasal dari *Firebase* ataupun dari *Rest API*.

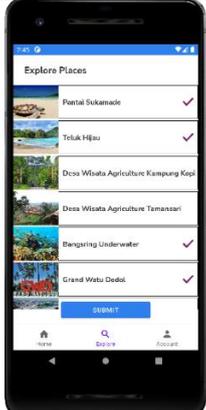
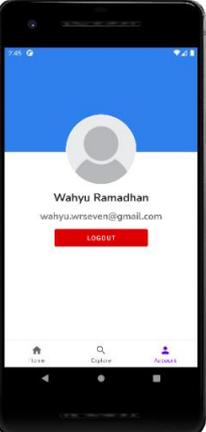
5. *Model* bertanggung jawab untuk menyimpan data riwayat hasil pencarian pengguna ke dalam *database* local agar pengguna dapat mengaksesnya tanpa melalui koneksi internet.
6. *Remote DataSource* bertanggung jawab untuk melakukan request data kepada API dengan memanfaatkan *library Retrofit*. *Request* yang dilakukan dengan metode POST dengan mengirim data melalui form berupa *text* yang berisi *list* lokasi berupa id lokasi, *latitude* dan *longitude* yang kemudian API akan memberikan *response* dengan mengirimkan data hasil nilai *return* berupa total jarak dan *list* rute.

### 3.3. Implementasi

Tahap implementasi merupakan tahap untuk merealisasikan desain yang telah dibuat. Pada penelitian ini penulis menggunakan bahasa pemrograman Kotlin untuk mengembangkan aplikasi Android dengan *pattern* MAD. Adapun tampilan aplikasi dapat dilihat pada Tabel 1.

**Tabel 1.** Tampilan Aplikasi

No	Halaman	Tampilan	Penjelasan
1	<i>Splash Screen</i>		Saat pengguna menjalankan aplikasi maka akan ditampilkan halaman <i>splash screen</i> sebagai sambutan awal kepada pengguna.
2	<i>Register</i>		Saat pengguna belum memiliki akun, maka pengguna diharuskan untuk melakukan registrasi agar dapat mengakses aplikasi dan dapat melakukan <i>login</i> . Pada halaman ini pengguna diminta untuk mengisi data berupa nama, email, kata sandi dan konfirmasi kata sandi.
3	<i>Login</i>		Saat pengguna telah memiliki akun, maka pengguna diharuskan untuk melakukan <i>login</i> agar dapat mengakses aplikasi. Pada halaman ini pengguna diminta untuk mengisi data berupa email dan kata sandi yang telah didaftarkan.

4	Beranda		<p>Setelah melakukan proses <i>login</i>, pengguna akan diarahkan ke <i>main menu</i> bagian <i>home</i>. Halaman ini menampilkan lokasi wisata yang populer dan menampilkan berita-berita tentang tempat wisata di Banyuwangi dan sekitarnya</p>
5	Halaman Eksplor (Pemilihan List Rute yang akan Dioptimasi)		<p>Halaman <i>explore</i> pada bagian kedua dari <i>main menu</i> menampilkan <i>list</i> semua data lokasi wisata. Pengguna dapat memilih beberapa lokasi wisata untuk dilakukan perhitungan agar mendapatkan rute optimal.</p>
6	Akun		<p>Halaman <i>account</i> adalah halaman terakhir dari <i>main menu</i>. Halaman ini memuat informasi <i>login</i> dari <i>user</i> dan halaman yang digunakan untuk melakukan <i>logout</i>.</p>
7	Hasil Optimasi Rute		<p>Halaman terakhir adalah halaman <i>maps</i> dimana halaman ini memuat hasil perhitungan dari <i>input user</i> yang berupa total jarak, rute, dan juga titik-titik lokasi pada <i>maps</i>.</p>

Selain tampilan aplikasi, penulis menggunakan algoritme *Ant Colony Optimization (ACO)* yang diimplementasikan pada *web service* sesuai dengan *development pattern* yang dapat dilihat pada Gambar 3. Algoritme ACO pada *web service* dibangun dengan menggunakan Bahasa pemrograman Python dan algoritme mampu untuk memproses *input user* yang berupa *list latitude* dan *longitude* dan mengembalikan hasil nilai berupa total jarak dan rute optimal yang dapat dilalui oleh pengguna.

### 3.4. Pengujian

Pengujian bertujuan untuk mengetahui apakah fungsional aplikasi dapat berjalan dengan baik dan untuk mengetahui persepsi dari pengguna terhadap aplikasi yang dikembangkan dengan cara memberikan kuesioner yang berisi penilaian perspektif kepada pengguna. Untuk menguji fungsional sistem penulis menggunakan *black box* dimana Pengujian *Black Box* adalah pengujian yang bertujuan untuk menunjukkan fungsi perangkat lunak [8]. Adapun hasil dari *black box testing* dapat dilihat pada Tabel 2.

**Tabel 2.** Hasil Pengujian Black Box

No	Pengujian	Hasil
1.	Melakukan proses <i>register</i>	Valid
2.	Melakukan proses <i>login</i>	Valid
3.	Mengambil dan menampilkan data lokasi populer dari	Valid
4.	Melakukan klik pada item berita dan menampilkannya	Valid
5.	Mengambil dan menampilkan seluruh data lokasi populer	Valid
6.	Melakukan seleksi item lokasi dan mengirimkan kepada server	Valid
7.	Menampilkan informasi <i>login</i> dari pengguna	Valid

Dari hasil pengujian fungsional yang telah dilakukan. Dapat disimpulkan bahwa aplikasi optimasi rute di Banyuwangi dapat beroperasi dengan baik.

Pengujian selanjutnya merupakan pengujian UAT. Pengujian UAT adalah pengujian yang dirancang khusus untuk memodelkan penerimaan pengguna terhadap suatu sistem informasi [9]. Dalam melakukan pengujian ini, terdapat 10 orang responden yang akan memberi penilaian terhadap aplikasi yang sudah dibangun. Pertanyaan yang akan diberikan kepada pengguna yang sebagai berikut:

1. Apakah aplikasi mudah untuk dioperasikan?
2. Apakah tata letak tampilan aplikasi jelas dan mudah diakses?
3. Apakah alur aplikasi mudah dipahami?
4. Apakah informasi hasil perhitungan optimasi rute sudah jelas?
5. Apakah informasi hasil perhitungan optimasi rute membantu Anda untuk mencari rute terdekat yang harus dilalui pada tiap-tiap destinasi wisata?
6. Apakah fitur detail pada aplikasi membantu Anda untuk mendapatkan informasi tentang destinasi wisata yang ingin dituju?
7. Secara umum, apakah tidak terdapat kendala saat Anda menggunakan aplikasi?
8. Apakah Anda akan menggunakan aplikasi ini untuk mencari informasi dan melakukan optimasi rute destinasi wisata di Banyuwangi di masa mendatang?

Terdapat lima kriteria Skala Likert yang digunakan oleh pengguna untuk melakukan penilaian dalam aplikasi. Skala Likert adalah suatu skala psikometrik yang umum digunakan dalam kuesioner, dan merupakan skala yang paling banyak digunakan dalam riset berupa survei [10]. Lima skala Likert sebagai kriteria penilaian dapat dilihat pada Tabel 3.

**Tabel 3.** Kriteria penilaian

Pernyataan	Nilai
Sangat Setuju	A
Setuju	B
Netral	C
Tidak Setuju	D
Sangat Tidak Setuju	E

Dari Tabel 3 dapat disimpulkan bahwa standar tertinggi dari skala Likert adalah Sangat Setuju atau dengan Nilai A, dan standar terendah adalah Sangat Tidak Setuju atau Nilai E.

Hasil dari pengujian UAT yang telah dilakukan dapat dilihat pada Tabel 4.

**Tabel 4.** Hasil Pengujian UAT

No	Pertanyaan	Persentase (%)				
		A	B	C	D	E
1	Apakah aplikasi mudah untuk dioperasikan?	60	30	10	0	0
2	Apakah tata letak tampilan aplikasi jelas dan mudah diakses?	70	20	10	0	0
3	Apakah alur aplikasi mudah dipahami?	50	50	0	0	0
4	Apakah informasi hasil perhitungan optimasi rute sudah jelas?	40	50	10	0	0
5	Apakah informasi hasil perhitungan optimasi rute membantu Anda untuk mencari rute terdekat yang harus dilalui pada tiap-tiap destinasi wisata?	60	30	10	0	0
6	Apakah fitur detail pada aplikasi membantu Anda untuk mendapatkan informasi tentang destinasi wisata yang ingin dituju?	30	60	10	0	0
7	Secara umum, apakah tidak terdapat kendala saat Anda menggunakan aplikasi?	80	10	10	0	0
8	Apakah Anda akan menggunakan aplikasi ini untuk mencari informasi dan melakukan optimasi rute destinasi wisata di Banyuwangi di masa mendatang?	30	50	20	0	0
<b>Rata-rata</b>		<b>52,5</b>	<b>37,5</b>	<b>10</b>	<b>0</b>	<b>0</b>

Dari Tabel 4 dapat dilihat bahwa aplikasi yang telah dibangun mendapatkan nilai rata-rata 52,5% responden sangat setuju, 37,5% responden setuju, lalu sisanya netral sehingga dapat disimpulkan bahwa 90% responden setuju bahwa aplikasi yang dikembangkan dapat menjadi sarana untuk wisatawan untuk melakukan optimasi rute dengan tampilan aplikasi yang menarik yang memiliki tampilan *user interface* yang mudah dipahami.

#### 4. Kesimpulan

Dari penelitian yang telah dilakukan, didapatkan hasil penelitian berupa aplikasi Android untuk melakukan optimasi rute. Hasil evaluasi sistem yaitu *Black Box* dan *User Acceptance Test* pada aplikasi menunjukkan bahwa fungsional sistem dapat bekerja dengan baik tanpa mengalami *bug* atau masalah pada setiap fitur dan halaman yang tersedia. Aplikasi yang dibangun cukup mendapat respon positif dari responden yang bersedia membantu untuk menguji aplikasi yang dibangun berdasarkan pertanyaan dan kriteria penilaian yang telah ditentukan. Selain itu, *Modern Android Development pattern* juga dapat diterapkan dengan sangat baik pada pengembangan aplikasi Android untuk optimasi rute destinasi wisata di Banyuwangi menggunakan algoritme *Ant Colony Optimization*.

#### Referensi

- [1] B. Santosa dan T.J. Ai, *Pengantar Metaheuristik: Implementasi dengan Matlab*, Surabaya: ITS Tekno Sains, 2017.
- [2] F. Nugroho, *Tutorial Netlogo Tutorial Pembuatan Game Dengan Bahasa Pemrograman Java*, Jakarta: Direktorat Pendidikan Tinggi Islam, 2012.
- [3] Android Developers, "Modern Android Development", <https://developer.android.com/modern-android-development> (diakses pada 12 Juli 2022).
- [4] A. Hendardi, "Penerapan Algoritma Floyd-Warshall dalam Layanan Informasi berbasis Web untuk Pencarian Lintasan terpendek Antar Program Studi/Fakultas di UPN Veteran Jawa Timur", Jawa Timur, Indonesia: Universitas Pembangunan Nasional Veteran, 2012.
- [5] M. D. A. Cipta Hasibuan and L. -, "Pencarian Rute Terbaik Pada Travelling Salesman Problem (TSP) Menggunakan Algoritma Genetika pada Dinas Kebersihan dan

- Pertamanan Kota Pekanbaru,” *SATIN - Sains dan Teknol. Inf.*, vol. 1, no. 1, p. 35, 2016, doi: 10.33372/stn.v1i1.11.
- [6] I. Manggolo, M. I. Marzuki, and M. Alaydrus, “Optimalisasi Perencanaan Jaringan Akses Serat Optik Fiber To The Home Menggunakan Algoritma Genetika,” *J. Telekomun. dan Komput.*, vol. 2, no. 1, p. 21, 2017, doi: 10.22441/incomtech.v2i1.1102.
- [7] Dicoding, “Modern Android Development”, <https://www.dicoding.com/academies/14/tutorials/19557> (diakses pada 12 Juli 2022).
- [8] T. Hidayat, dan M. Muttaqin, “Pengujian Sistem Informasi Pendaftaran dan Pembayaran Wisuda Online menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis”, *Jurnal Teknik Informatika UNIS (JUTIS)*, vol. 6, pp.25-29, 2018, doi: 10.33592/jutis.Vol6.Iss1.38.
- [9] S.N.A. Fahmi, “Penerapan Technology Acceptance Test dalam Pengujian Sistem Informasi Sarana dan Prasarana Sekolah di MTs Negeri 5 Kabupaten Kediri”, Malang: UIN Maulana Malik Ibrahim, 2019.
- [10] E. Suwandi, F. H. Imansyah, and H. Dasril, “Analisis Tingkat Kepuasan Menggunakan Skala Likert pada Layanan Speedy yang Bermigrasi ke Indihome,” *J. Tek. Elektro*, p. 11, 2018.

*This page is intentionally left blank.*