

Implementasi Algoritma Haversine dan K-Nearest Neighbors (KNN) pada Aplikasi Pencarian Kos Berbasis Android Menggunakan Flutter

Khoirul Mustofa¹, Ahmad Heru Mujiyanto²

^{1,2}Fakultas Teknologi Informasi, Teknik Informatika, Universitas Hasyim Asy'ari Tebuireng, Jombang, Indonesia

Informasi Artikel

Diterima Redaksi: 25 Juni 2025
Revisi Akhir: 20 Juli 2025
Diterbitkan Online: 06 November 2025

Kata Kunci

Pencarian kos; Haversine; K-Nearest Neighbors; Flutter; Android

Korespondensi

Phone:
E-mail:
khoirulmustofa@mhs.unhasy.ac.id

ABSTRAK

Pencarian kos yang efisien dan akurat menjadi kebutuhan penting bagi masyarakat, khususnya mahasiswa dan pekerja perantau. Penelitian ini mengusulkan solusi digital berupa aplikasi pencarian kos berbasis Android yang dibangun menggunakan framework Flutter. Aplikasi ini mengimplementasikan algoritma Haversine untuk menghitung jarak geografis antara pengguna dan lokasi kos, serta algoritma K-Nearest Neighbors (KNN) untuk merekomendasikan kos terdekat berdasarkan parameter tertentu seperti harga, fasilitas, dan jarak. Data kos disimpan dalam basis data Firebase, dan sistem menggunakan GPS untuk menentukan posisi pengguna secara real-time. Hasil pengujian kuantitatif dan fungsional menunjukkan bahwa kombinasi kedua algoritma ini memberikan peningkatan signifikan pada kualitas rekomendasi. Implementasi Algoritma Haversine menunjukkan tingkat akurasi yang tinggi, di mana hasil perhitungan jaraknya memiliki tingkat kesamaan mendekati 98% bila dibandingkan dengan data rute dari Google Maps. Sementara itu, Algoritma KNN yang diterapkan untuk sistem rekomendasi terbukti 100% berhasil dalam mengurutkan dan menyajikan daftar kos berdasarkan kedekatan lokasi sesuai dengan skenario pengujian Alpha. Dengan demikian, aplikasi ini mampu memberikan rekomendasi yang tidak hanya akurat secara geografis tetapi juga relevan secara fungsional, dan diharapkan dapat menjadi solusi praktis bagi pengguna dalam menemukan kos yang sesuai dengan preferensi mereka.

1. PENDAHULUAN

Perkembangan teknologi informasi yang pesat telah memberikan pengaruh besar pada berbagai sektor, termasuk kebutuhan akan informasi yang cepat dan akurat dalam sektor perumahan. Salah satu tantangan yang sering dihadapi oleh masyarakat, terutama mahasiswa dan pekerja perantau, adalah proses pencarian tempat tinggal atau kos yang sesuai dengan kebutuhan mereka. Cakupan wilayah yang luas serta banyaknya pilihan yang tersedia sering kali membuat pencarian menjadi lebih sulit, memakan waktu, dan membutuhkan usaha lebih. Selain itu, informasi mengenai fasilitas, harga, dan kondisi lingkungan seringkali tidak terstruktur dengan baik, sehingga pengguna kesulitan menemukan kos yang sesuai kriteria. Tingginya mobilitas di kalangan ini juga menuntut adanya sistem pencarian yang dapat memberikan hasil real-time dan akurat.

Seiring dengan kemajuan teknologi, aplikasi mobile telah menjadi solusi efisien untuk berbagai permasalahan sehari-hari, termasuk dalam pencarian kos. Namun, tantangan utama dalam pengembangan aplikasi semacam ini adalah bagaimana menyajikan hasil pencarian yang relevan berdasarkan lokasi pengguna. Untuk mengatasi masalah ini, penelitian ini mengusulkan sebuah aplikasi pencarian kos berbasis Android yang dibangun menggunakan framework Flutter. Solusi ini dirancang dengan mengimplementasikan dua algoritma utama. Pertama, algoritma Haversine digunakan untuk menghitung jarak geografis antara dua titik di permukaan bumi dengan mempertimbangkan kelengkungan bumi, sehingga mampu memberikan estimasi jarak yang lebih akurat antara pengguna dan lokasi kos. Kedua, algoritma K-

Nearest Neighbors (KNN) diterapkan sebagai metode klasifikasi untuk mengelompokkan dan merekomendasikan kos yang paling relevan berdasarkan kedekatan jarak serta preferensi pengguna lainnya.

Beberapa penelitian sebelumnya telah mengeksplorasi penggunaan algoritma ini secara terpisah. Riset oleh Rohim (2023) menerapkan algoritma KNN untuk merekomendasikan kos, namun tidak mengintegrasikannya dengan perhitungan jarak spesifik seperti Haversine. Di sisi lain, penelitian oleh Hasanah, dkk (2023) dan Sidiq, dkk (2022) telah mengimplementasikan algoritma Haversine untuk fitur geolokasi dan pendeteksian lokasi, tetapi tidak menggabungkannya dengan algoritma rekomendasi seperti KNN. Penelitian lain oleh Husada, dkk (2020) juga memanfaatkan Haversine Formula untuk menentukan jarak terdekat ke fasilitas publik, namun fokusnya adalah perbandingan hasil dengan tool lain, bukan untuk sistem rekomendasi properti.

Keterbaruan dalam penelitian ini terletak pada integrasi kedua algoritma tersebut Haversine untuk akurasi jarak dan KNN untuk relevansi rekomendasi dalam sebuah aplikasi mobile yang dibangun dengan framework Flutter. Kombinasi ini diharapkan dapat memberikan hasil pencarian yang lebih optimal dibandingkan metode konvensional atau yang hanya menggunakan salah satu algoritma. Penelitian ini bertujuan untuk menghasilkan sebuah aplikasi pencarian kos berbasis Android yang dikembangkan menggunakan framework Flutter. Fokus utama dari penelitian ini adalah mengimplementasikan dan menguji efektivitas dua algoritma inti: Algoritma Haversine yang berfungsi untuk menghitung jarak geografis secara akurat, serta Algoritma KNearest Neighbors (KNN) untuk membangun sistem rekomendasi kos berdasarkan kedekatan lokasi.

2. METODE PENELITIAN

2.1 Tahapan Penelitian

Penelitian ini menggunakan model pengembangan perangkat lunak Waterfall yang memiliki alur kerja sistematis dan berurutan. Tahapan dalam model ini meliputi analisis kebutuhan, perancangan sistem, implementasi, pengujian, dan pemeliharaan. Alur penelitian diawali dengan identifikasi masalah, yaitu kesulitan dalam mencari informasi kos yang akurat. Selanjutnya dilakukan studi literatur untuk mengkaji penelitian relevan dan menentukan metode yang akan digunakan.

Gambar 1: Model Pengembangan Sistem

Pengumpulan data sebagai bahan penelitian dilakukan melalui tiga teknik utama: wawancara dengan beberapa pemilik kos untuk mengetahui detail fasilitas yang ditawarkan, observasi langsung ke lokasi untuk memvalidasi data, serta pengambilan data koordinat geografis dari Google Maps. Objek penelitian ini adalah aplikasi pencarian kos yang berfokus pada perhitungan jarak dan rekomendasi. Perangkat keras yang digunakan untuk membangun dan menguji aplikasi meliputi laptop Asus A409FJ dengan prosesor Intel Core i5 Gen 8 dan RAM 20 GB, serta sebuah ponsel pintar Xiaomi M4 Pro. Sementara itu, perangkat lunak yang digunakan terdiri dari sistem operasi Ubuntu 24.04, Visual Studio Code, Android Studio, dan framework Flutter dengan bahasa pemrograman Dart. Untuk pengelolaan basis data, penelitian ini memanfaatkan Firebase sebagai Backend as a Service (BaaS).

Teknik analisis data dalam penelitian ini terbagi menjadi dua proses utama yang saling terintegrasi. Pertama, algoritma Haversine diterapkan untuk menganalisis data lokasi. Algoritma ini digunakan untuk menghitung jarak antara dua titik koordinat geografis (lokasi pengguna dan lokasi kos) dengan memperhitungkan kelengkungan permukaan bumi. Rumus Haversine mengolah data latitude dan longitude untuk menghasilkan jarak dalam satuan kilometer (km), yang menjadi

dasar untuk menentukan kedekatan sebuah kos dari posisi pengguna. Kedua, setelah jarak untuk setiap kos berhasil dihitung, algoritma K-Nearest Neighbors (KNN) digunakan untuk memproses hasil tersebut. KNN berperan untuk mengklasifikasikan dan mengurutkan daftar kos berdasarkan jarak terpendek dari lokasi pengguna. Sistem akan merekomendasikan sejumlah 'k' kos terdekat, sehingga menyajikan hasil yang paling relevan bagi pengguna. Tahap akhir dari metode ini adalah pengujian Alpha dengan teknik Black Box untuk memastikan bahwa seluruh fungsionalitas aplikasi, terutama implementasi kedua algoritma, dapat berjalan sesuai dengan hasil yang diharapkan.

2.2 Algoritma Haversine

Menurut (Dimas Dwi Prihantoro, 2022) Haversine Formula merupakan algoritma yang digunakan untuk menghitung suatu jarak dengan mengambil kedua titik permukaan bumi yang digunakan dalam suatu rute dan memberikan suatu jarak lingkaran besar dengan mengambil longitude latitude sebagai inputannya. Maka latitude dan longitude dapat memberikan informasi jarak dari kedua titik tersebut.

Haversine adalah persamaan penting dalam sistem navigasi, nantinya formula Haversine ini akan menghasilkan jarak terpendek antara dua titik, misalnya pada bola yang diambil dari garis bujur (longitude) dan garis lintang (latitude). Hukumnya adalah semua persamaan yang digunakan berdasarkan bentuk bumi yang bulat (spherical earth) dengan menghilangkan faktor bahwa bumi itu sedikit elips (elipsoidal factor). Ini merupakan formula umum dalam trigonometri bola yang berkaitan dengan sisi dan sudut segitiga bola (Suzuki Syofian & Agam Arian Damar. 2020).

2.3 Algoritma K-Nearest Neighbors (KNN)

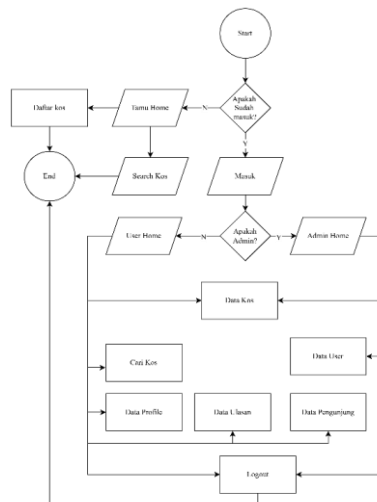
Algoritma K-Nearest Neighbors merupakan metode klasifikasi yang mengelompokkan data baru berdasarkan jarak data/tetangga (neighbors) terdekat. Algoritma menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari query instance yang baru. Algoritma bekerja berdasarkan jarak terpendek dari query instance ke sampel data latih untuk menentukan. Sampel data latih diproyeksikan ke ruang berdimensi banyak, Dimana masing-masing dimensi merepresentasikan fitur dari data. Sebuah titik pada ruang ini ditandai jika merupakan klasifikasi yang paling banyak ditemukan pada K buah tetangga terdekat dari titik tersebut (Tyo angin virnando. 2022). Algoritma nearest neighbor adalah pendekatan untuk pencarian masalah memakailah relasi antara kasus baru dengan kasus lampau, yaitu bersumber pada kecocokan bobot sebanyak fitur yang ada (Aries Diw Indriyanti. 2020).

3. HASIL DAN PEMBAHASAN

Hasil dari penelitian ini adalah sebuah aplikasi pencarian kos berbasis Android yang berhasil dibangun menggunakan framework Flutter. Aplikasi ini secara fungsional mengimplementasikan algoritma Haversine dan K-Nearest Neighbors (KNN) untuk menyediakan fitur pencarian dan rekomendasi kos. Seluruh data aplikasi, seperti data pengguna, data kos, dan ulasan, dikelola menggunakan basis data Firebase.

3.1 Perancangan Sistem

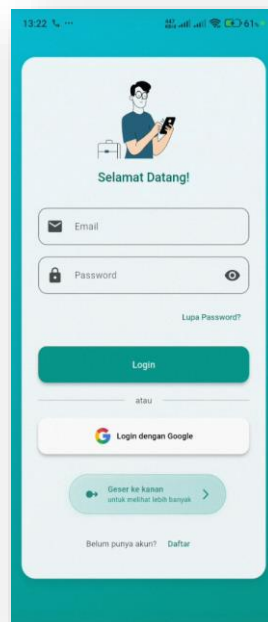
Untuk merancang alur kerja aplikasi, digunakan sebuah flowchart atau diagram alir seperti pada (Gambar 2). Diagram ini menjelaskan proses dari awal hingga akhir. Saat pengguna membuka aplikasi, sistem akan mengecek apakah ia sudah punya akun. Jika belum, pengguna harus mendaftar dulu. Jika sudah, ia bisa langsung login. Setelah berhasil masuk, sistem membedakan alur untuk dua jenis pengguna: user biasa dan admin. User biasa akan diarahkan ke menu utama untuk mencari kos, melihat profil, dan melakukan booking. Sementara itu, admin memiliki halaman khusus untuk mengelola data data penting seperti data kos dan data pengguna. Proses bagi keduanya akan selesai saat mereka keluar dari aplikasi (logout).



Gambar 2: Flowchart Sistem Pencarian Kos

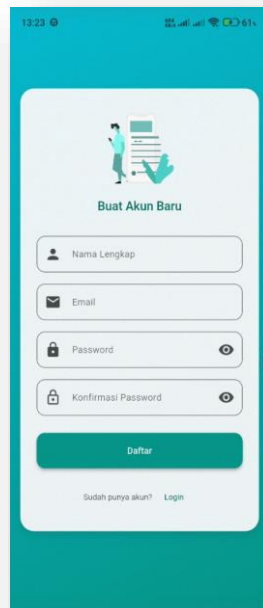
3.2 Implementasi Interface

Implementasi antarmuka (interface) merupakan tahap realisasi dari perancangan desain yang telah disusun sebelumnya dalam sebuah alur pengembangan yang terstruktur (Pratama & Daru, 2022). Dengan menggunakan framework Flutter, seluruh antarmuka aplikasi dikembangkan untuk platform Android, di mana framework ini mempermudah proses pembuatan antarmuka menjadi produk fungsional (Saputra & Wibowo, 2021). Perancangan antarmuka ini berfokus untuk menciptakan pengalaman pengguna (User Experience) yang baik dan mudah dipahami, serta mendukung efektivitas dan efisiensi penggunaan aplikasi yang dapat diukur melalui analisis usability (Setiawan & Anam, 2022). Pada bagian ini, akan dipaparkan hasil implementasi dari setiap halaman utama aplikasi, mulai dari halaman pembuka, autentikasi pengguna, hingga halaman fungsional lainnya.



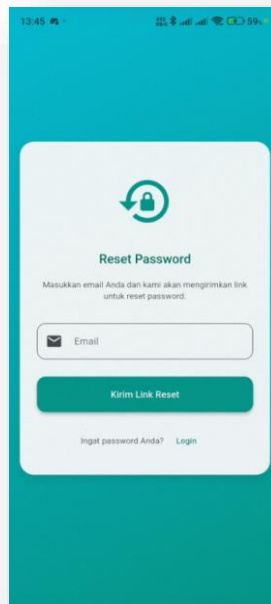
Gambar 3: Halaman Login

Halaman ini merupakan gerbang autentikasi bagi pengguna terdaftar, seperti pemilik kos dan admin. Sistem menyediakan dua metode masuk: menggunakan email dan password, atau melalui akun Google sebagai alternatif. Halaman ini juga menyediakan navigasi ke fitur pendaftaran dan pemulihan password.



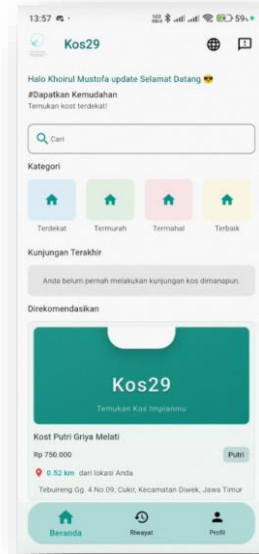
Gambar 3: Halaman Pendaftaran

Halaman ini berfungsi untuk registrasi akun baru bagi pengguna. Pengguna diwajibkan mengisi data diri berupa nama lengkap, alamat email, dan password untuk membuat akun. Setelah pendaftaran berhasil, pengguna dapat masuk ke sistem untuk mengelola data atau mencari kos.



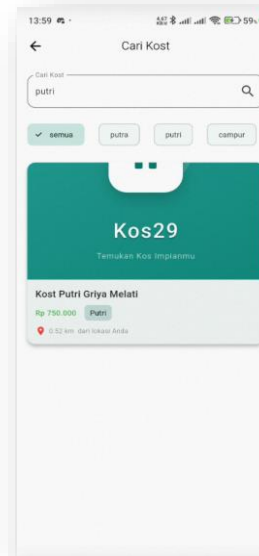
Gambar 5: Halaman Lupa Password

Fitur ini dirancang untuk membantu pengguna memulihkan akses ke akun mereka jika lupa kata sandi. Pengguna perlu memasukkan alamat email yang terdaftar, dan sistem akan mengirimkan tautan untuk proses pengaturan ulang (reset) password.



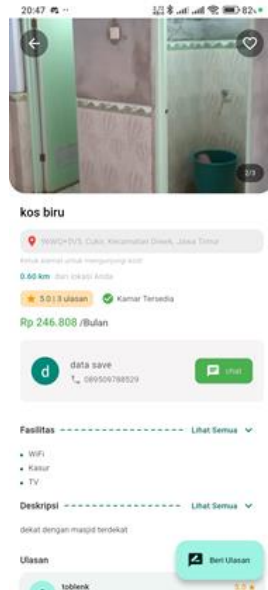
Gambar 4: Halaman Home

Halaman ini berfungsi sebagai dasbor utama aplikasi. Fitur utamanya adalah kolom pencarian, kategori pencarian cepat, riwayat kunjungan terakhir, serta daftar rekomendasi kos. Kategori "Terdekat" dan bagian "Direkomendasikan" merupakan hasil langsung dari implementasi algoritma K-Nearest Neighbors (KNN) yang mengurutkan kos berdasarkan jarak terpendek dari lokasi pengguna.



Gambar 5: Halaman Pencarian

Halaman ini menampilkan hasil pencarian kos berdasarkan kata kunci dan filter yang diterapkan pengguna. Setiap item hasil menyajikan informasi penting seperti nama kos, harga, dan jarak akurat dari lokasi pengguna. Informasi jarak ini merupakan output dari kalkulasi menggunakan algoritma Haversine.

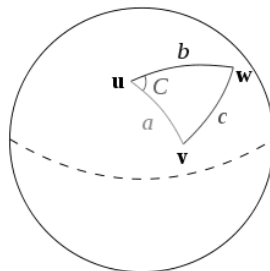


Gambar 8: halaman Detail Kos

Halaman ini menyajikan informasi rinci dari satu properti kos. Data yang ditampilkan mencakup foto, alamat, harga, ketersediaan kamar, fasilitas, deskripsi, serta ulasan dari pengguna lain. Salah satu informasi kunci pada halaman ini adalah jarak presisi dari lokasi pengguna, yang dihitung secara real-time menggunakan algoritma Haversine untuk membantu pengguna dalam mengambil keputusan.

3.3 Implementasi Algoritma Haversine

Implementasi algoritma Haversine berhasil menghitung jarak geografis antara lokasi pengguna yang dideteksi melalui GPS dengan koordinat setiap kos yang tersimpan di database. Hasil perhitungan jarak menunjukkan tingkat akurasi yang tinggi.



Gambar 6: Model Bola Dunia

Model Bola Dunia menampilkan model geometris bola dunia yang menjadi dasar perhitungan jarak menggunakan algoritma Haversine. Model ini mengasumsikan bumi sebagai sebuah bola sempurna, sehingga jarak terpendek antara dua titik dalam penelitian ini adalah lokasi pengguna dan lokasi kos dapat dihitung sebagai panjang busur pada lingkaran besar (great-circle distance). Segitiga bola yang terbentuk dengan sisi a, b, dan c merepresentasikan hubungan spasial antara dua titik koordinat. Dengan menerapkan hukum Haversine dari trigonometri bola pada model ini, koordinat lintang (latitude) dan bujur (longitude) dapat diolah untuk menghasilkan estimasi jarak yang akurat dengan memperhitungkan kelengkungan bumi.

$$a = \sin^2 \left(\frac{\Delta \text{lat}}{2} \right) + \cos(\text{lat}_1) \cdot \cos(\text{lat}_2) \cdot \sin^2 \left(\frac{\Delta \text{long}}{2} \right)$$

$$d = R \cdot c$$

Berdasarkan pengujian yang membandingkan hasil perhitungan rumus Haversine dengan jarak yang ditampilkan oleh Google Maps, ditemukan bahwa selisihnya sangat kecil, sehingga membuktikan bahwa algoritma ini efektif dan dapat diandalkan untuk menentukan jarak akurat pada aplikasi. Tabel 1 menunjukkan contoh hasil perhitungan jarak menggunakan rumus Haversine dari satu titik lokasi pengguna ke beberapa lokasi kos yang berbeda.

Setelah jarak dihitung, algoritma K-Nearest Neighbors (KNN) diimplementasikan untuk mengurutkan data kos berdasarkan jarak terdekat. Hasilnya ditampilkan kepada pengguna dalam bentuk daftar rekomendasi yang telah terurut, di mana kos dengan jarak terpendek muncul di posisi teratas.

Rumus KNN:

$$d(p, q) = \sqrt{\sum_{i=1}^n (q_i - p_i)^2}$$

d(p,q): Jarak Euclidean antara titik **p** dan titik **q**.

n: Jumlah dimensi atau fitur data (misalnya: harga, jarak, rating fasilitas).

p,q: Dua titik data yang akan dihitung jaraknya.

pi,qi: Nilai dari fitur ke-**i** pada titik **p** dan **q**.

Tabel 1: Hasil Perhitungan Jarak dengan Haversine dan KNN

| Nama Kos | Koordinat Pengguna (Lat, Long) | Koordinat Kos (Lat, Long) | Jarak (Haversine) |
|------------------|--------------------------------|---------------------------|-------------------|
| Green Kos Baru | -7.605418, 112.234457 | -7.603983, 112.232218 | 0.31 km |
| putri malu kos | -7.605418, 112.234457 | -7.603964, 112.229403 | 0.59 km |
| kos mhs barat | -7.605418, 112.234457 | -7.608050, 112.241393 | 0.81 km |
| kos mandiri mami | -7.605418, 112.234457 | -7.614389, 112.226344 | 1.34 km |

Fungsionalitas seluruh fitur aplikasi kemudian divalidasi melalui pengujian Alpha dengan metode Black Box. Hasil pengujian menunjukkan bahwa semua skenario uji, mulai dari otentikasi pengguna, manajemen data, hingga fungsi pencarian dan rekomendasi, berjalan dengan sukses dan sesuai dengan yang diharapkan.

Tabel 2: Pengujian Alpha halaman Login

| Skenario Uji | Hasil yang Diharapkan | Hasil Uji |
|--|--|-----------|
| Pengguna memasukkan email yang valid dan password yang valid, lalu menekan tombol login. | Pengguna berhasil login dan diarahkan ke halaman utama aplikasi. | Berhasil |
| Pengguna memasukkan email yang valid dan password yang salah, lalu menekan tombol login. | Muncul pesan kesalahan yang memberitahukan bahwa password salah. | Berhasil |
| Pengguna memasukkan email yang salah dan password yang valid, lalu menekan tombol login. | Muncul pesan kesalahan yang memberitahukan bahwa email salah. | Berhasil |
| Pengguna memilih opsi "Lupa Password?". | Pengguna diarahkan ke halaman pemulihan password. | Berhasil |
| Pengguna menekan tombol "Login dengan Google". | Pengguna diarahkan ke halaman login Google. | Berhasil |
| Pengguna menekan tombol "Daftar" untuk membuat akun baru. | Pengguna diarahkan ke halaman pendaftaran akun baru. | Berhasil |

Tabel 3: Pengujian Alpha halaman Pendaftaran

| Skenario Uji | Hasil yang Diharapkan | Hasil Uji |
|---|--|-----------|
| Mengisi semua kolom dengan data valid dan klik "Daftar" | Akun baru berhasil dibuat dan pengguna diarahkan ke halaman berikutnya | Berhasil |
| Mengisi kolom email dengan format salah dan klik "Daftar" | Muncul pesan error bahwa email tidak valid | Berhasil |
| Mengisi password dan konfirmasi password yang tidak sama lalu klik "Daftar" | Muncul pesan error bahwa password tidak cocok | Berhasil |
| Mengosongkan salah satu kolom lalu klik "Daftar" | Muncul pesan error bahwa semua kolom wajib diisi | Berhasil |
| Menekan tombol "Login" di bagian bawah | Dialihkan ke halaman login | Berhasil |

Tabel 4: Pengujian Alpha halaman Lupa password

| Skenario Uji | Hasil yang Diharapkan | Hasil Uji |
|---|--|-----------|
| Memasukkan email valid lalu klik "Kirim Link Reset" | Akun baru berhasil dibuat dan pengguna diarahkan ke halaman berikutnya | Berhasil |
| Memasukkan email tidak valid lalu klik "Kirim Link Reset" | Muncul pesan error bahwa email tidak valid | Berhasil |
| Mengosongkan kolom email lalu klik "Kirim Link Reset" | Muncul pesan error bahwa password tidak cocok | Berhasil |
| Menekan link "Login" di bagian bawah | Muncul pesan error bahwa semua kolom wajib diisi | Berhasil |

3.4 Pembahasan

Pembahasan dari hasil penelitian ini menunjukkan bahwa tujuan penelitian telah tercapai. Implementasi algoritma Haversine berhasil menyediakan perhitungan jarak yang akurat, yang merupakan fondasi penting untuk fitur pencarian berbasis lokasi. Selanjutnya, penerapan algoritma KNN secara efektif memanfaatkan hasil perhitungan jarak tersebut untuk memberikan rekomendasi yang relevan kepada pengguna, sehingga mempermudah dan mempercepat proses pengambilan keputusan. Temuan ini sejalan dengan penelitian sebelumnya yang memanfaatkan Haversine untuk geolokasi dan KNN untuk sistem rekomendasi, namun keunggulan penelitian ini terletak pada integrasi keduanya untuk menciptakan sistem yang lebih sinergis dan akurat. Kombinasi kedua algoritma ini terbukti mampu memberikan rekomendasi lokasi kos yang lebih relevan dan akurat dibandingkan metode pencarian konvensional yang tidak didasari oleh perhitungan jarak matematis. Dengan demikian, aplikasi yang dikembangkan tidak hanya berfungsi sebagai direktori, tetapi juga sebagai sistem pendukung keputusan yang cerdas bagi para pencari kos.

4. KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan yang telah diuraikan, dapat ditarik beberapa kesimpulan. Pertama, implementasi Algoritma Haversine dalam aplikasi pencarian kos ini telah berhasil menghitung jarak antara lokasi pengguna dan lokasi kos secara akurat dengan memanfaatkan data koordinat latitude dan longitude. Algoritma ini terbukti efektif sebagai dasar untuk menentukan posisi kos terdekat yang dapat dijadikan rujukan oleh pengguna. Kedua, penerapan Algoritma K-Nearest Neighbors (KNN) berhasil mengurutkan dan merekomendasikan daftar kos berdasarkan jarak terdekat yang telah dihitung sebelumnya. Dengan pendekatan ini, sistem mampu menyajikan daftar kos yang paling relevan secara efisien tanpa membebani pengguna dengan proses pencarian manual. Dengan demikian, integrasi kedua algoritma tersebut terbukti mampu menciptakan sebuah solusi yang efektif dan efisien untuk mempermudah pengguna dalam menemukan tempat tinggal yang sesuai dengan preferensi lokasi mereka.

DAFTAR PUSTAKA

- Hasanah, M., & Suharto, A. (2023). Algoritma Haversine Pada Sistem Informasi Geografis: Tinjauan Literatur Sistematis. *Nuansa Informatika*, 17(2), 173-181.
- Husada, C., dkk. (2020). Implementasi Haversine Formula Untuk Pembuatan Sig Jarak Terdekat Ke Rs Rujukan Covid-19. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 4(5), 851-859.

- Layliyah, A., Indriyanti, A. D., & Wiratsongko, R. (2020). Diagnosa Penyakit Tanaman Jagung Dengan Metode Case Based Reasoning Berbasis Android. *Inovate: Jurnal Ilmiah Inovasi Teknologi Informasi*, 5(1).
- Pratama, D. F. W., & Daru, A. F. (2022). Penerapan Metode Waterfall untuk Pengembangan Sistem Informasi Perpustakaan Berbasis Web Menggunakan Framework CodeIgniter. *Information Science and Library*, 3(1), 55-66.
- Prihantoro, D. D. (2022). Implementasi Algoritma Haversine Formula Dan Location Based Service Pada Aplikasi Pencarian Lokasi Bird Contest Berbasis Android. *Jurnal Media Informatika Budidarma*, 6(3), 1373-1380.
- Rohim, M. Y. A. (2023). *Implementasi Algoritma K-Nearest Neighbor Untuk Mencari Kos Dikota Palembang Berbasis Android*. Universitas Muhammadiyah Palembang.
- Saputra, A. D., & Wibowo, A. (2021). Perancangan dan Implementasi User Interface dan User Experience Aplikasi Pemesanan Makanan “Mamayo” Menggunakan Flutter. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 5(10), 4227-4236.
- Setiawan, B., & Anam, C. (2022). Analisis Usability Aplikasi Mobile E-Complaint Menggunakan System Usability Scale (SUS) Berbasis Framework Flutter. *Jurnal Media Informatika Budidarma*, 6(2), 995-1002.
- Sidiq, H., dkk. (2022). Implementasi Algoritma Haversine Pada Absensi Kepegawaian Berbasis Android. *Journal Of Applied Computer Science And Technology (JACOST)*, 3(1), 58-64.
- Syofian, S., & Damar, A. A. (2020). *Implementasi Algoritma First Come First Served Dan Haversine Pada Aplikasi Pemesanan Makanan Berbasis Mobile*. Repository.Unsada.Ac.Id.
- Virnando, T. A. (2022). Penerapan Algoritma K-Nearest Neighbor (K-Nn) Pada Klasifikasi Kualitas Hasil Pengeringan Bunga Cengkih. *Jurnal Rekayasa Teknologi Dan Komputasi*, 5(1), 22-29.
- Prastyo, E. H. A., Prisma, I. P. E., & Wiratsongko, R. (2020). Implementasi Web Scraping Pada Situs Berita Menggunakan Metode Supervised learning. *Inovate: Jurnal Ilmiah Inovasi Teknologi Informasi*, 5(1), 58-66.
- Prehanto, D. R., Indriyanti, A. D., Prisma, I., Permadi, G. S., & Prastyo, E. H. A. (2021). Implementation of Web Scraping on News Sites Using the Supervised Learning Method. *Ilkogretim Online*, 20(3).
- Mashuri, C., Prastyo, E. H. A., & Hariri, F. R. *Improving Fake News Detection Accuracy with Lexicon-based Approach and LSTM through Text Preprocessing*. *Jurnal Sistem Informasi Bisnis*, 15(2).
- Prastyo, E. (2024). Deteksi berita hoax dengan pendekatan *Lexicon Based dan LSTM* (Doctoral dissertation, Universitas Islam Negeri Maulana Malik Ibrahim).
- Prastyo, E. H. A., Yaqin, M. A., Faisal, M., & Firdaus, R. A. J. (2024). *Naive Bayes Classification for Software Defect Prediction*. *Transactions on Informatics and Data Science*, 1(1), 11-20.