IMPLEMENTASI ALGORITMA *SPHERICAL TRIGONOMETRY* DAN *SOLAR POSITION* UNTUK PENENTUAN ARAH KIBLAT PADA APLIKASI *MOBILE OFFLINE* BERBASIS FLUTTER

Muhammad Nur Falak Minhar Lukman, Herdiesel Santoso*
Informatika, Sekolah Tinggi Manajemen Informatika dan Ilmu Komputer Yogyakarta
Jl. Sisingamangaraja No.76, Brontokusuman, Kec. Mergangsan, Daerah Istimewa Yogyakarta
herdiesel.santoso@stmikelrahma.ac.id

ABSTRAK

Penentuan arah kiblat merupakan kebutuhan penting dalam pelaksanaan ibadah salat, terutama dalam menentukan orientasi bangunan masjid maupun posisi individu saat beribadah. Berbagai aplikasi penentu arah kiblat berbasis *smartphone* telah dikembangkan untuk mempermudah pengguna, namun sebagian besar masih bergantung pada sensor magnetometer yang rentan terhadap gangguan medan magnet sehingga dapat menyebabkan deviasi arah. Penelitian ini bertujuan untuk mengimplementasikan algoritma *spherical trigonometry* dalam aplikasi mobile berbasis Flutter untuk menghitung arah kiblat secara matematis berdasarkan koordinat geografis pengguna. Metode penelitian menggunakan model *Rapid Application Development* (RAD) dengan integrasi modul komputasi lokal sehingga aplikasi dapat berjalan tanpa koneksi internet. Selain perhitungan azimut kiblat, sistem juga mengimplementasikan metode *solar position* untuk menghitung azimut Matahari sebagai mekanisme validasi terhadap arah yang dihasilkan. Hasil pengujian pada lima lokasi berbeda di Indonesia menunjukkan bahwa aplikasi mampu menghasilkan nilai azimut kiblat pada rentang 291°–295° sesuai variasi koordinat geografis. Implementasi sistem menunjukkan bahwa perhitungan berbasis *spherical trigonometry* dapat memberikan hasil arah kiblat yang konsisten dan tidak bergantung pada sensor magnetometer perangkat.

Kata kunci : *Kiblat, Spherical Trigonometry, Solar Position, Aplikasi Mobile, RAD*

1. PENDAHULUAN

Penentuan arah kiblat merupakan aspek penting dalam pelaksanaan ibadah salat bagi umat Islam, baik dalam menentukan orientasi bangunan masjid maupun posisi individu saat beribadah. Secara ilmiah, arah kiblat dapat dihitung sebagai sudut azimut dari suatu lokasi menuju Ka'bah dengan memanfaatkan koordinat lintang dan bujur pada sistem koordinat geografis. Salah satu pendekatan yang banyak digunakan dalam perhitungan tersebut adalah *spherical trigonometry* yang memodelkan hubungan sudut antar titik pada permukaan bumi melalui konsep segitiga bola sehingga arah menuju Ka'bah dapat dihitung secara matematis [1], [2].

Seiring perkembangan teknologi mobile, berbagai aplikasi penentu arah kiblat berbasis *smartphone* telah dikembangkan untuk mempermudah masyarakat memperoleh informasi arah kiblat secara cepat dan praktis. Namun sebagian besar aplikasi tersebut masih mengandalkan sensor magnetometer atau kompas digital perangkat yang rentan terhadap gangguan medan magnet di lingkungan sekitar. Kondisi ini berpotensi menyebabkan deviasi arah sehingga hasil yang ditampilkan tidak selalu stabil atau akurat [3]. Selain itu, beberapa penelitian juga menunjukkan bahwa perbedaan metode perhitungan serta ketergantungan pada sensor perangkat dapat menghasilkan variasi nilai azimut antar aplikasi yang digunakan oleh masyarakat [4], [5].

Untuk mengatasi permasalahan tersebut, penentuan arah kiblat dapat dilakukan menggunakan pendekatan perhitungan matematis berbasis koordinat geografis sehingga tidak bergantung pada sensor magnetik perangkat. Metode *spherical trigonometry* memungkinkan arah menuju Ka'bah dihitung secara deterministik berdasarkan parameter lintang dan bujur pengguna sehingga hasil perhitungan bersifat konsisten selama data koordinat valid [6]. Selain itu, validasi arah juga dapat dilakukan menggunakan pendekatan astronomis melalui perhitungan posisi Matahari (*solar position*) yang menghasilkan informasi azimut Matahari berdasarkan waktu dan lokasi tertentu [7]. Algoritma perhitungan posisi Matahari yang dipublikasikan oleh NOAA memungkinkan parameter tersebut dihitung secara komputasional sehingga dapat dimanfaatkan sebagai mekanisme verifikasi terhadap sistem penentuan arah kiblat [8].

Pengembangan aplikasi mobile yang mengintegrasikan algoritma komputasi tersebut dapat dilakukan menggunakan framework Flutter yang mendukung pengembangan aplikasi lintas platform dengan satu basis kode [9]. Agar proses pengembangan sistem berjalan secara sistematis dan efisien, penelitian ini menggunakan pendekatan *Rapid Application Development* (RAD) yang memungkinkan pembangunan aplikasi dilakukan secara iteratif dan terstruktur [10].

Berdasarkan latar belakang tersebut, penelitian ini bertujuan untuk mengimplementasikan algoritma *spherical trigonometry* dalam aplikasi mobile berbasis Flutter untuk menghitung arah kiblat berdasarkan

koordinat geografis pengguna. Sistem yang dikembangkan juga mengintegrasikan perhitungan *solar position* sebagai mekanisme validasi terhadap hasil perhitungan arah kiblat sehingga aplikasi diharapkan mampu memberikan hasil perhitungan yang konsisten secara matematis serta dapat digunakan secara offline tanpa ketergantungan pada sensor magnetometer perangkat.

2. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Beberapa penelitian sebelumnya telah membahas metode komputasi dalam penentuan arah kiblat. Walhidayah dan Ismail mengimplementasikan rumus segitiga bola untuk menentukan azimut kiblat menggunakan pemrograman Python dan menunjukkan bahwa pendekatan *spherical trigonometry* mampu menghasilkan perhitungan arah yang konsisten berdasarkan koordinat geografis [1]. Penelitian lain juga menunjukkan bahwa metode *spherical trigonometry* dapat digunakan secara efektif dalam aplikasi komputasi untuk menentukan arah kiblat dengan tingkat akurasi yang tinggi [2], [6]. Selain itu, penelitian yang membandingkan berbagai metode dan instrumen penentuan arah kiblat menunjukkan bahwa pendekatan geometri bola memiliki tingkat kesesuaian yang baik ketika dibandingkan dengan metode astronomi lainnya maupun sistem digital yang digunakan dalam aplikasi arah kiblat [11], [12]. Penelitian terkait juga mengkaji penggunaan metode Vincenty dalam perhitungan arah kiblat dengan mempertimbangkan model bumi berbentuk elipsoid untuk meningkatkan presisi perhitungan [13]. Hasil evaluasi terhadap aplikasi arah kiblat berbasis *smartphone* juga menunjukkan bahwa penggunaan sensor magnetometer dapat menyebabkan deviasi arah akibat gangguan medan magnet di lingkungan sekitar [3], [4]. Oleh karena itu, pendekatan berbasis perhitungan matematis menjadi alternatif yang lebih stabil dalam implementasi sistem penentuan arah kiblat berbasis komputasi.

2.2. *Spherical Trigonometry* dalam Penentuan Arah Kiblat

Perhitungan arah kiblat secara matematis dapat dilakukan menggunakan algoritma *spherical trigonometry*, yaitu metode trigonometri yang diterapkan pada permukaan bola untuk menghitung relasi sudut antar titik koordinat geografis [1]. Dalam konteks ini, azimut kiblat dihitung sebagai sudut dari arah utara geografis menuju Ka'bah berdasarkan nilai lintang dan bujur suatu lokasi.

Implementasi *spherical trigonometry* memungkinkan perhitungan azimut dilakukan secara deterministik dengan memanfaatkan fungsi trigonometri seperti sinus, cosinus, dan tangen untuk memperoleh sudut arah secara presisi [6]. Pendekatan ini banyak digunakan dalam sistem komputasi karena relatif efisien dan tidak memerlukan sensor tambahan selama koordinat lokasi tersedia. Penelitian sebelumnya menunjukkan bahwa perhitungan berbasis *spherical trigonometry* mampu menghasilkan nilai azimut yang stabil dan konsisten untuk berbagai lokasi pengujian [7]. Penelitian lain juga menunjukkan bahwa metode *spherical trigonometry* memberikan hasil arah kiblat yang konsisten dan memiliki tingkat kesesuaian yang tinggi ketika dibandingkan dengan algoritma astronomi lain maupun sistem penentuan arah kiblat berbasis digital [11], [14].

Selain *spherical trigonometry*, terdapat pendekatan lain seperti metode *Vincenty* yang mempertimbangkan model bumi yang lebih kompleks dalam proses perhitungan arah dan jarak antar dua titik koordinat [13]. Meskipun metode tersebut memberikan pendekatan yang lebih detail, implementasinya cenderung lebih kompleks dibandingkan *spherical trigonometry* yang lebih sederhana dan sesuai untuk aplikasi *mobile*.

2.3. *Solar Position* sebagai Validasi Perhitungan

Selain menghitung azimut kiblat, sistem komputasi modern juga dapat mengintegrasikan metode *solar position* untuk menentukan posisi Matahari berdasarkan waktu dan koordinat geografis tertentu. Perhitungan *solar position* menghasilkan nilai azimut Matahari dan sudut elevasi yang dapat digunakan untuk menganalisis arah bayangan pada bidang horizontal [7]. Hubungan sudut antara azimut Matahari dan arah bayangan memberikan dasar verifikasi konsistensi matematis terhadap sistem perhitungan yang diterapkan.

Metode *solar position* memungkinkan validasi internal dilakukan tanpa memerlukan alat ukur tambahan karena seluruh parameter dihitung secara komputasional berdasarkan waktu dan lokasi [1]. Dengan demikian, integrasi *spherical trigonometry* dan *solar position* dalam satu sistem memberikan pendekatan ganda, yaitu perhitungan arah kiblat dan pengecekan konsistensi sudut melalui posisi Matahari.

Evaluasi terhadap aplikasi penentu arah kiblat menunjukkan bahwa sistem yang hanya mengandalkan sensor magnetik memiliki potensi deviasi akibat gangguan lingkungan sekitar [3], [15]. Oleh karena itu, pendekatan berbasis perhitungan matematis seperti *spherical trigonometry* yang didukung validasi *solar position* menjadi alternatif yang lebih stabil untuk diimplementasikan dalam aplikasi digital.

2.4. Flutter dalam Implementasi Aplikasi *Mobile*

Pengembangan aplikasi *mobile* yang mengintegrasikan algoritma komputasi memerlukan *framework* yang mendukung efisiensi dan kompatibilitas lintas platform. Flutter merupakan framework yang memungkinkan pengembangan aplikasi dengan satu basis kode untuk berbagai sistem operasi secara efisien [16]. Keunggulan ini mendukung implementasi modul perhitungan *spherical trigonometry* dan *solar position* secara lokal pada perangkat pengguna.

Pendekatan pengembangan sistem yang terstruktur diperlukan agar integrasi antara modul komputasi dan antarmuka pengguna berjalan dengan baik [10]. Dalam konteks pengembangan aplikasi arah kiblat, arsitektur sistem harus mampu mengelola proses pengambilan koordinat, perhitungan azimut, serta visualisasi hasil secara stabil dan responsif. Integrasi melalui komponen WebView memungkinkan proses komputasi dilakukan secara lokal sehingga aplikasi tetap dapat beroperasi dalam kondisi *offline*.

Beberapa penelitian juga menunjukkan bahwa pengembangan aplikasi *mobile* berbasis layanan keagamaan dapat dilakukan secara efektif melalui pendekatan rekayasa perangkat lunak yang sistematis [17]. Dengan demikian, implementasi *spherical trigonometry* dan *solar position* dalam aplikasi berbasis Flutter memerlukan desain sistem yang jelas agar hasil perhitungan dapat ditampilkan secara akurat dan konsisten.

3. METODE PENELITIAN

Berdasarkan landasan teori mengenai implementasi *spherical trigonometry* dan integrasi *solar position* dalam sistem komputasi *mobile*, diperlukan tahapan penelitian yang sistematis untuk merealisasikan algoritma tersebut ke dalam aplikasi yang dapat digunakan secara langsung oleh pengguna. Metode penelitian ini menjelaskan pendekatan yang digunakan dalam perancangan sistem, implementasi algoritma, serta proses pengujian dan validasi hasil perhitungan secara terstruktur.

3.1. Jenis dan Pendekatan Penelitian

Penelitian ini termasuk dalam kategori penelitian terapan (*applied research*) yang berfokus pada implementasi algoritma matematis ke dalam sistem aplikasi *mobile*. Penelitian terapan bertujuan menghasilkan solusi praktis melalui pemanfaatan teori dan metode yang telah ada untuk menyelesaikan permasalahan nyata [10], [18]. Dalam konteks ini, algoritma *spherical trigonometry* dan metode *solar position* diimplementasikan dalam aplikasi berbasis Flutter untuk menghasilkan sistem penentuan arah kiblat yang dapat digunakan secara langsung oleh pengguna.

Pendekatan yang digunakan dalam penelitian ini adalah pendekatan rekayasa perangkat lunak, yaitu proses pengembangan sistem yang dilakukan secara bertahap mulai dari analisis kebutuhan hingga tahap pengujian. Pendekatan ini memungkinkan integrasi modul perhitungan matematis dengan antarmuka pengguna dilakukan secara sistematis sehingga sistem yang dibangun memiliki struktur arsitektur yang jelas [9]. Dengan pendekatan tersebut, implementasi algoritma tidak hanya diuji dari sisi komputasi, tetapi juga dari sisi fungsionalitas aplikasi secara keseluruhan.

3.2. Model Pengembangan Sistem

Model pengembangan sistem yang digunakan dalam penelitian ini adalah *Rapid Application Development* (RAD). RAD merupakan model pengembangan perangkat lunak yang menekankan pada proses pembangunan sistem secara cepat melalui pendekatan iteratif dan prototyping [19]. Model ini memungkinkan pengembang untuk menghasilkan versi awal sistem dalam waktu singkat, kemudian melakukan perbaikan dan penyempurnaan secara bertahap berdasarkan hasil evaluasi.

RAD terdiri dari beberapa tahapan utama, yaitu perencanaan kebutuhan, desain sistem, konstruksi atau implementasi, serta pengujian dan perbaikan sistem [20]. Pada tahap perencanaan kebutuhan, dilakukan identifikasi fitur utama aplikasi seperti pengambilan koordinat GPS, perhitungan azimut kiblat menggunakan *spherical trigonometry*, serta perhitungan azimut Matahari menggunakan metode *solar position*. Tahap desain sistem mencakup perancangan arsitektur aplikasi dan alur proses perhitungan dari input koordinat hingga keluaran hasil.

Tahap konstruksi dilakukan dengan mengimplementasikan algoritma *spherical trigonometry* dan *solar position* ke dalam aplikasi berbasis Flutter melalui integrasi modul komputasi lokal. Proses ini dilakukan secara iteratif sehingga setiap modul yang telah dibangun dapat langsung diuji untuk memastikan kesesuaian fungsi dan akurasi perhitungan. Tahap terakhir adalah pengujian dan penyempurnaan sistem, yaitu proses evaluasi terhadap hasil perhitungan dan stabilitas aplikasi sebelum digunakan sebagai sistem final.

Pemilihan model RAD dalam penelitian ini didasarkan pada kebutuhan pengembangan aplikasi yang relatif terfokus pada implementasi algoritma komputasi dengan siklus perbaikan yang cepat. Dengan pendekatan ini, proses pengembangan dapat dilakukan secara fleksibel namun tetap terstruktur sehingga integrasi antara modul perhitungan dan antarmuka pengguna dapat berjalan secara optimal.

3.3. Perancangan Arsitektur Sistem

Perancangan arsitektur sistem dilakukan untuk menggambarkan alur proses komputasi dari input hingga menghasilkan output berupa azimut kiblat dan informasi posisi Matahari. Arsitektur sistem dirancang agar modul perhitungan *spherical trigonometry* dan *solar position* dapat terintegrasi secara terstruktur dalam aplikasi berbasis Flutter.

Secara umum, sistem terdiri dari empat komponen utama, yaitu modul pengambilan koordinat, modul perhitungan *spherical trigonometry*, modul perhitungan *solar position*, dan modul antarmuka pengguna. Modul pengambilan koordinat berfungsi untuk memperoleh data lintang dan bujur pengguna melalui layanan GPS perangkat. Data koordinat ini menjadi parameter utama dalam proses perhitungan azimut kiblat dan posisi Matahari.

Modul *spherical trigonometry* bertanggung jawab menghitung azimut kiblat berdasarkan koordinat pengguna dan koordinat Ka'bah. Perhitungan dilakukan secara lokal dengan menggunakan fungsi trigonometri yang telah diimplementasikan dalam skrip komputasi. Selanjutnya, modul *solar position* menghitung azimut dan elevasi Matahari berdasarkan waktu dan lokasi pengguna. Hasil dari kedua modul tersebut kemudian dikirim ke modul antarmuka untuk ditampilkan dalam bentuk nilai numerik pada layar aplikasi.

Integrasi modul komputasi dilakukan melalui komponen WebView pada Flutter sehingga proses perhitungan dapat berjalan secara lokal tanpa memerlukan koneksi internet. Pendekatan ini memastikan bahwa aplikasi tetap dapat berfungsi dalam kondisi *offline* dan tidak bergantung pada layanan eksternal. Arsitektur sistem yang modular juga memudahkan proses pengujian karena setiap modul dapat diuji secara terpisah sebelum diintegrasikan secara keseluruhan.

3.4. Implementasi Algoritma *Spherical Trigonometry*

Perhitungan arah kiblat pada penelitian ini diimplementasikan menggunakan pendekatan *spherical trigonometry*, yaitu metode trigonometri pada permukaan bola yang umum digunakan dalam navigasi dan penentuan arah pada koordinat geografis [6], [21]. Pendekatan ini memodelkan bumi sebagai bola dan menghitung relasi sudut antara lokasi pengguna dan Ka'bah melalui dua tahap perhitungan, yaitu penentuan jarak sudut dan penentuan azimut kiblat.

Tahap pertama adalah menghitung jarak sudut d antara lokasi pengguna dan Ka'bah menggunakan hukum cosinus segitiga bola (*Law of Cosines for Sides*) pada persamaan (1) berikut:

$$\cos d = \sin\phi_1 \sin\phi_2 + \cos\phi_1 \cos\phi_2 \cos(\Delta\lambda) \quad (1)$$

Keterangan,

- ϕ_1 : Lintang geografis lokasi pengguna (radian)
- ϕ_2 : Lintang geografis Ka'bah (radian)
- λ_1 : Bujur lokasi pengguna (radian)
- λ_2 : Bujur Ka'bah (radian)
- $\Delta\lambda = \lambda_2 - \lambda_2$: Selisih bujur
- d : Sudut pusat (*central angle*) yang dibentuk oleh lokasi pengguna dan Ka'bah terhadap pusat bumi

Nilai d merepresentasikan sudut antara dua vektor radius bumi yang ditarik dari pusat bumi menuju lokasi pengguna dan Ka'bah. Dalam implementasi komputasional, seluruh parameter sudut dinyatakan dalam radian karena fungsi trigonometri seperti sinus, cosinus, dan invers cosinus bekerja dalam satuan radian. Sebelum dilakukan perhitungan trigonometri, seluruh koordinat lintang dan bujur dikonversi dari satuan derajat ke radian menggunakan persamaan (2).

$$\theta_{rad} = \theta_{\deg} \times \frac{\pi}{180} \quad (2)$$

Konversi ini diperlukan karena fungsi trigonometri dalam sistem komputasi menggunakan satuan radian. Setelah diperoleh nilai jarak sudut d, azimut kiblat dihitung menggunakan turunan hukum cosinus segitiga bola untuk sudut pada persamaan (3) berikut:

$$\cos A = \frac{\sin\phi_1 \cos d}{\sin d \cos\phi_1} \quad (3)$$

Keterangan,

- A : Azimut kiblat terhadap utara geografis
- d : Jarak sudut lokasi pengguna dan Ka'bah terhadap pusat bumi
- ϕ_1 : Lintang lokasi pengguna
- ϕ_2 : Lintang Ka'bah

Nilai azimut yang diperoleh kemudian dikonversi ke satuan derajat dan disesuaikan dengan sistem azimut navigasi yang diukur dari arah utara searah jarum jam. Tanda selisih bujur digunakan dalam implementasi untuk menentukan orientasi arah Timur atau Barat sehingga sudut berada pada kuadran yang sesuai [7].

Pendekatan *spherical trigonometry* yang digunakan bersifat deterministik karena hanya bergantung pada parameter lintang dan bujur tanpa dipengaruhi sensor magnetik perangkat, sehingga hasil perhitungan tetap konsisten selama data koordinat valid [6].

3.5. Implementasi *Solar Position*

Sistem mengimplementasikan algoritma *solar position* berdasarkan *General Solar Position Calculations* yang dipublikasikan oleh *NOAA Global Monitoring Division* [8] sebagai mekanisme validasi internal terhadap hasil perhitungan arah kiblat. Masukan yang digunakan meliputi tanggal, waktu lokal, lintang, bujur, dan zona waktu pengguna yang diperoleh secara otomatis dari perangkat.

Proses perhitungan diawali dengan menentukan sudut tahunan pada persamana (4).

$$\gamma = \frac{2\pi}{365}(DOY - 1) \quad (4)$$

Nilai γ digunakan untuk menghitung deklinasi matahari (δ) dan *equation of time* (E_t). Berdasarkan kedua parameter tersebut, sistem menghitung *True Solar Time* (TST) dengan mengoreksi waktu lokal terhadap bujur dan zona waktu, yang kemudian dikonversi menjadi *hour angle* (H) dalam radian. Sudut zenit Matahari dihitung menggunakan persamaan (5) [8]:

$$\cos Z = \sin\phi \sin\delta + \cos\phi \cos\delta \cos H \quad (5)$$

Keterangan,

- Z : Sudut zenit Matahari (radian)
- ϕ : Lintang lokasi pengguna (radian)
- δ : Deklinasi Matahari (radian)
- H : *Hour angle* terhadap meridial lokal (radian)

Setelah sudut zenit dikonversi ke derajat, elevasi Matahari dihitung menggunakan persamaan (6).

$$h = 90° - Z \quad (6)$$

Azimut Matahari dihitung menggunakan persamaan (7) [8].

$$\cos\theta = \frac{-(\sin\phi\cos Z - \sin\delta)}{\cos\phi\sin Z} \quad (7)$$

Keterangan,

- θ : Azimut Matahari diukur searah jarum jam dari utara sejati (derajat)
- Z : Sudut zenit Matahari (radian)
- ϕ : Lintang lokasi pengguna (radian)
- δ : Deklinasi Matahari (radian)

Penyesuaian kuadran dilakukan berdasarkan nilai hour angle pada persamaan (8).

$$\theta_{sun} = \begin{cases} \theta, & H < 0 \\ 360° - \theta, & H \geq 0 \end{cases} \quad (8)$$

Nilai θ_{sun} ditampilkan sebagai pointer Matahari pada antarmuka kompas digital. Sistem memvalidasi arah kiblat dengan membandingkan pointer kiblat hasil *spherical trigonometry* dan pointer Matahari dari metode *solar position* secara visual. Pada kondisi rashdul qiblat, kedua pointer berimpit sehingga pengguna dapat melakukan verifikasi mandiri menggunakan bayangan benda [4].

4. HASIL DAN PEMBAHASAN

4.1. Perancangan Sistem Menggunakan UML

Perancangan sistem pada penelitian ini menggunakan *Unified Modeling Language* (UML) sebagai alat bantu untuk memodelkan interaksi dan alur kerja aplikasi. Mengingat fokus penelitian terletak pada implementasi algoritma penentuan arah kiblat berbasis pendekatan astronomis, maka pemodelan UML dibatasi pada diagram yang merepresentasikan perilaku sistem, yaitu Use Case Diagram dan Activity Diagram.

Pendekatan ini dipilih agar pemodelan tetap proporsional dan tidak bergeser ke dokumentasi teknis implementasi perangkat lunak, melainkan berfungsi sebagai representasi konseptual alur kerja aplikasi.

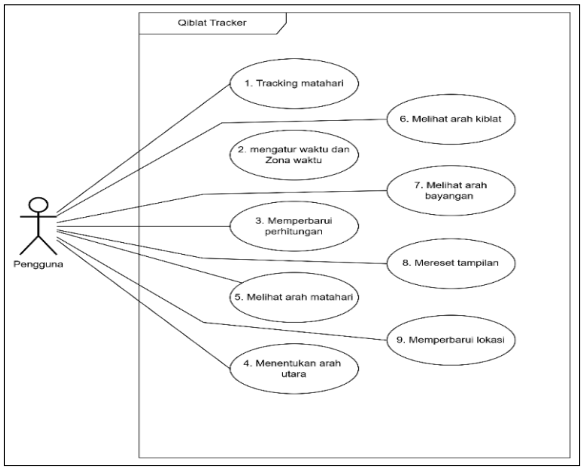
4.2. Use Case Diagram

Use Case Diagram digunakan untuk menggambarkan interaksi antara pengguna dengan sistem aplikasi *mobile* penentu arah kiblat. Pada sistem ini hanya terdapat satu aktor utama, yaitu Pengguna. Pengguna berinteraksi langsung dengan aplikasi melalui fitur-fitur utama yang tersedia pada halaman utama aplikasi.

Fungsi-fungsi utama sistem meliputi:

a. Mengakses lokasi GPS
b. Memulai pelacakan posisi Matahari (*Track*)
c. Menghentikan pelacakan (*Stop*)
d. Menghitung posisi Matahari berdasarkan waktu manual (*Calc*)
e. Menampilkan arah utara sejati dan arah kiblat (*North*)

Use case pada gambar 1 menggambarkan bahwa seluruh proses penentuan arah kiblat berpusat pada interaksi tunggal antara pengguna dan sistem tanpa melibatkan sistem eksternal, karena aplikasi dirancang untuk bekerja secara *offline*.



Gambar 1. Use case diagram aplikasi

4.3. Activity Diagram

Activity Diagram digunakan untuk menggambarkan alur kerja sistem secara keseluruhan, mulai dari aplikasi dijalankan hingga proses validasi arah kiblat selesai dilakukan. Activity diagram dapat di lihat pada gambar 2.

Alur aktivitas sistem adalah sebagai berikut:

a. Aplikasi dijalankan
b. Sistem meminta izin akses lokasi
c. Sistem memperoleh koordinat lintang dan bujur dari GPS
d. Sistem menghitung azimut kiblat menggunakan Persamaan (1) hingga persamaan (3)
e. Pengguna memilih mode operasi

Jika pengguna menekan tombol *Track*:

a. Sistem menghitung posisi Matahari secara kontinu menggunakan Persamaan (4) hingga persamaan (8)
b. Azimut Matahari dan arah bayangan diperbarui secara real-time

Jika pengguna menekan tombol *Stop*:

a. Proses pelacakan dihentikan
b. Nilai terakhir posisi Matahari dipertahankan

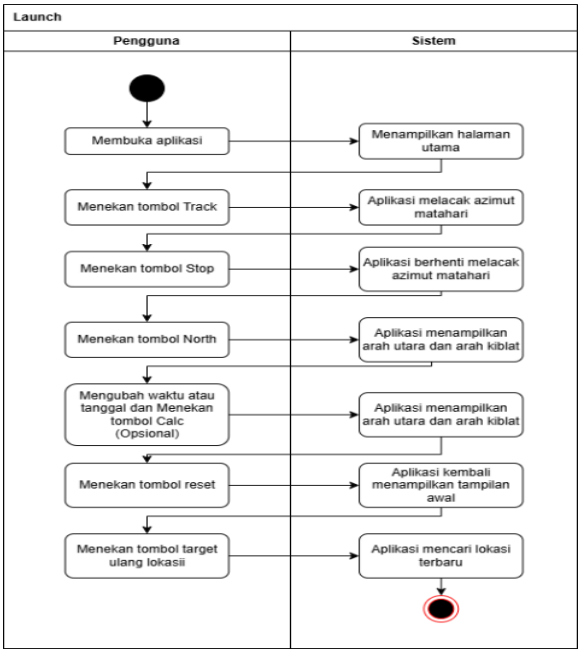
Jika pengguna mengisi waktu manual dan menekan tombol *Calc*:

a. Sistem menghitung ulang posisi Matahari berdasarkan waktu input menggunakan Persamaan (4) hingga persamaan (8)
b. Hasil ditampilkan tanpa mengaktifkan tracking

Jika pengguna menekan tombol *North*:

a. Sistem menampilkan arah utara sejati
b. Sistem menampilkan arah kiblat berdasarkan hasil perhitungan azimut

Proses ini menunjukkan bahwa seluruh perhitungan astronomis berjalan di dalam sistem tanpa memerlukan koneksi internet, karena data falak telah disimpan secara lokal.



Gambar 2. Activity diagram aplikasi

4.4. Hasil Implementasi Sistem

Aplikasi Qiblat Tracker diimplementasikan menggunakan Flutter dengan integrasi modul komputasi berbasis WebView yang menjalankan perhitungan *spherical trigonometry* dan *solar position* secara lokal pada perangkat. Sistem tidak memerlukan koneksi internet karena seluruh proses komputasi dilakukan secara *offline*.

Ketika aplikasi dijalankan, sistem meminta izin akses lokasi untuk memperoleh lintang (ϕ_1) dan bujur (λ_1). Koordinat tersebut dikonversi dari derajat ke radian menggunakan persamaan (2) sebelum digunakan dalam proses trigonometri.

Perhitungan arah kiblat dilakukan menggunakan persamaan (1) untuk memperoleh jarak sudut (d) antara lokasi pengguna dan Ka'bah. Nilai tersebut kemudian digunakan dalam persamaan (3) untuk menghitung azimut kiblat terhadap utara geografis. Nilai azimut kiblat bersifat tetap selama koordinat tidak berubah.

Modul *solar position* diaktifkan ketika pengguna menekan tombol *Track*. Pada kondisi ini, sistem menjalankan persamaan (4) hingga persamaan (8) secara kontinu berdasarkan waktu perangkat. Perhitungan dimulai dengan menentukan sudut tahunan (γ) berdasarkan DOY, kemudian menghitung deklinasi Matahari dan *equation of time*. Nilai tersebut digunakan untuk menentukan *hour angle*, sudut zenit (Z), elevasi Matahari, serta azimut Matahari. Selama mode pelacakan aktif, azimut Matahari dan arah bayangan diperbarui secara *real-time*.

Ketika tombol *Stop* ditekan, sistem menghentikan pembaruan *solar position* dan membekukan nilai azimut Matahari pada waktu tersebut. Pada tahap ini muncul indikator "*Sight to Sun*" yang digunakan untuk menyelaraskan perangkat dengan posisi Matahari aktual. Proses ini memungkinkan validasi arah kiblat menggunakan metode bayangan benda tegak lurus.

Setelah penyelarasan dilakukan, tombol *North* dapat ditekan untuk menampilkan arah utara sejati dan arah kiblat hasil perhitungan persamaan (3). Fitur ini hanya dapat digunakan ketika waktu sistem sesuai dengan kondisi aktual di lapangan.

Selain mode pelacakan *real-time*, aplikasi menyediakan tombol *Calc* yang berfungsi untuk melakukan simulasi perhitungan *solar position* berdasarkan waktu, tanggal, dan zona waktu yang dimasukkan secara manual. Fitur ini hanya dapat digunakan ketika mode pelacakan dalam keadaan berhenti.

Saat tombol *Calc* ditekan, sistem mengeksekusi kembali persamaan (4) hingga persamaan (8) berdasarkan parameter waktu yang diinput pengguna. Hasilnya berupa perubahan azimut Matahari dan arah bayangan pada dial kompas. Fitur ini memungkinkan pengguna mengamati arah azimut Matahari pada waktu lampau maupun waktu yang akan datang, termasuk kondisi malam hari ketika elevasi Matahari bernilai negatif.

Namun demikian, hasil perhitungan melalui tombol *Calc* tidak dapat digunakan untuk validasi arah utara sejati karena waktu simulasi tidak selalu sesuai dengan kondisi Matahari aktual di lapangan. Oleh karena itu, tombol *North* hanya dapat digunakan setelah proses validasi dilakukan menggunakan waktu nyata. Tampilan akhir aplikasi arah kiblat dan utara sejati ditunjukkan pada gambar 3.

Gambar 3. Hasil tampilan aplikasi saat menunjuk arah utara dan kiblat

4.5. Pengujian Perhitungan Arah Kiblat

Pengujian dilakukan pada lima lokasi berbeda dengan memanfaatkan fitur fake GPS untuk mensimulasikan perubahan koordinat. Tujuan pengujian ini adalah untuk menganalisis konsistensi algoritma *spherical trigonometry* dalam menghitung azimut kiblat berdasarkan variasi lintang dan bujur.

Parameter yang digunakan dalam pengujian meliputi:

a. Lintang lokasi (ϕ_1)
b. Bujur lokasi (λ_1)
c. Koordinat Ka'bah (ϕ_2 dan λ_2)
d. Waktu pengujian (tidak mempengaruhi azimut kiblat)

Table 1. Hasil Pengujian Perhitungan Azimut Kiblat

Lintang (LS/LU)	Bujur (BT)	Wilayah	Azimut Kiblat (°)
-7.846441	110.357822	Yogyakarta	294.73°
-6.913787	107.611720	Bandung	295.17°
-0.707197	119.855084	Palu	291.83°
-7.343595	112.773965	Surabaya	294.05°
-0.488808	117.123761	Balikpapan	292.00°

Berdasarkan tabel 1, terlihat bahwa nilai azimut kiblat berada pada $291° - 295°$ untuk wilayah Indonesia bagian tengah dan barat. Perbedaan nilai azimut disebabkan oleh variasi lintang dan bujur masing-masing lokasi terhadap posisi Ka'bah.

Wilayah yang lebih ke timur seperti Palu dan Balikpapan menunjukkan nilai azimut yang sedikit lebih kecil dibandingkan wilayah Jawa. Hal ini sesuai dengan karakteristik geometri bola, di mana perubahan bujur mempengaruhi sudut arah menuju titik referensi global.

Nilai azimut yang dihasilkan menunjukkan perubahan yang logis dan tidak bersifat acak, sehingga dapat disimpulkan bahwa implementasi algoritma *spherical trigonometry* dalam aplikasi berjalan secara konsisten terhadap variasi koordinat. Pengujian ini membuktikan bahwa sistem mampu menghitung arah kiblat secara stabil tanpa ketergantungan pada sensor magnetometer perangkat.

5. KESIMPULAN DAN SARAN

Penelitian ini menunjukkan bahwa penerapan algoritma *spherical trigonometry* dan *solar position* berhasil diimplementasikan dalam aplikasi *mobile* berbasis Flutter untuk menentukan arah kiblat secara *offline*. Perhitungan azimut kiblat berdasarkan geometri bola menghasilkan nilai yang konsisten terhadap variasi koordinat geografis pengguna, sementara integrasi *solar position* memungkinkan perhitungan deklinasi, *hour angle*, sudut zenit, dan azimut Matahari yang digunakan sebagai mekanisme validasi visual melalui metode bayangan. Kombinasi kedua pendekatan tersebut menghasilkan sistem yang mampu berjalan secara mandiri tanpa ketergantungan pada koneksi internet maupun sensor magnetometer secara penuh. Penelitian selanjutnya dapat mengembangkan perhitungan berbasis model elipsoid seperti metode *Vincenty* untuk membandingkan tingkat presisi dengan pendekatan *spherical trigonometry* serta melakukan pengujian komparatif guna meningkatkan akurasi dan keandalan sistem.

DAFTAR PUSTAKA

[1] R. Walhidayah and Ismail, "Penentuan Azimut Kiblat Masjid Kampus UIN Sultanah Nahrasiyah Lhokseumawe Menggunakan Rumus Segitiga Bola Berbasis Python," *Astroislamica: Journal of Islamic Astronomy*, vol. 4, no. 2, pp. 276–294, Dec. 2025, doi: 10.47766/astroislamica. v4i2.6012.

[2] Nursyahfira and L. M. Rizki, "Aplikasi Matematika dalam Penentuan Arah Kiblat dengan Menggunakan Spherical Trigonometry," *Jurnal Pendidikan Tambusai*, vol. 5, no. 3, pp. 114583–114591, 2021.

[3] S. O. Sriani and L. Ukhti, "Uji Akurasi Arah Kiblat Menggunakan Fitur Kompas Kiblat Pada Aplikasi Quran Kemenag Versi 2.1.4," *Astroislamica: Journal of Islamic Astronomy*, vol. 1, no. 2, pp. 213–231, Dec. 2022, doi: 10.47766/astroislamica.v1i2.951.

[4] R. Umar, W. J. Awan, N. A. Zaki, A. I. S. Izdinhar, and J. H. Azeez, "Accuracy of Qibla Direction: Evaluating Smartphone Apps with the Istiwa' Adzam Method," *Islamiyyat*, vol. 47, no. 1, pp. 3–14, Jun. 2025, doi: 10.17576/islamiyyat-2025-4701-01.

[5] N. N. Ismail, A. Musa, and O. Zainon, "The Accuracy of Qibla Direction Applications Used by the Public," *International Journal of Academic Research in Progressive Education and Development*, vol. 13, no. 4, pp. 2217–2224, Nov. 2024, doi: 10.6007/ijarped/v13-i4/23813.

[6] W. Uriawan, A. Nuralamsyah, A. Giffari, A. R. Mubarok, A. Muthmainnah, and N. N. S. Herdiyana, "Implementation of the Spherical Trigonometry Algorithm in a Qibla Direction Application," Bandung, Indonesia, Dec. 2025. doi: 10.48550/arXiv.2601.00808.

[7] D. Abdullah, "Determining the Qibla Direction by Astronomical and Geometrical Methods," Moscow, Russia, Dec. 2025. doi: https://doi.org/10.48550/arXiv.2512.03271.

[8] NOAA Global Monitoring Division, "General Solar Position Calculations," Boulder, Colorado, USA. Accessed: Feb. 26, 2026. [Online]. Available: https://gml.noaa.gov/grad/solcalc/

[9] Y. Widianto, Y. H. Pesik, and Y. Hari, "Perancangan dan Pengembangan Aplikasi Mobile Sales Order Berbasis Flutter pada Perusahaan Dua Jaya," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 5, pp. 10615–10622, Oct. 2024, doi: https://doi.org/ 10.36040/jati.v8i5.11097.

[10] G. R. Iriane and K. S. Japi, "Penerapan Metode RAD pada Perancangan Aplikasi Pegawai Berbasis Mobile Web pada Dinas Kelautan dan

Perikanan Provinsi NTT," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 9, no. 4, pp. 6131–6137, Aug. 2025, doi: https://doi.org/10.36040/jati.v9i4.13999.

[11] F. Y. Sari *et al.*, "Comparison of Spherical Trigonometry Method, Jean Meeus Algorithm and Google Qibla Finder in Determining of the Qibla Direction of Islamic Hospital," *Al-Hilal Journal of Islamic Astronomy*, vol. 5, no. 2, pp. 117–134, Oct. 2023, doi: 10.21580/al-hilal.2023.5.2.17192.

[12] M. K. Mubarraq, "Mizwala Qibla Finder: Modern Innovations in Qibla Direction Determination," *Al-Hisab: Journal of Islamic Astronomy*, vol. 2, no. 4, pp. 253–264, Dec. 2025, [Online]. Available: https://jurnal.umsu.ac.id/index.php/alhisab/article/view/27461

[13] M. Mundhir, "Implementasi Perhitungan Arah Kiblat Menggunakan Metode Vincenty dalam Bahasa Pemrograman C++," *Astroislamica: Journal of Islamic Astronomy*, vol. 4, no. 2, pp. 199–219, Dec. 2025, doi: 10.47766/astroislamica.v4i2.6005.

[14] C. L. Maknun, "Aplikasi Spherical Trigonometry dalam Menentukan Arah Kiblat Umat Islam," *Jurnal Fourier*, vol. 10, no. 2, pp. 57–66, Oct. 2021, doi: 10.14421/fourier.2021.102.57-66.

[15] Maulidin and Abdullah, "Uji Komparasi Instrumen Arah Kiblat Antara Qibla Tracker dan Mizwala Qibla Finder," *Astroislamica: Journal of Islamic Astronomy*, vol. 1, no. 1, pp. 73–96, Jun. 2022, doi: 10.47766/astroislamica.v1i1.899.

[16] I. Husain, Purwantoro, and Carudin, "Analisis Performa State Management Provider dan GetX pada Aplikasi Flutter," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 7, no. 2, pp. 1417–1422, Apr. 2023, doi: https://doi.org/10.36040/jati.v7i2.6867.

[17] F. Fathurahman, "Innovative Development of Mobile Application for Qibla Direction Guidance Services Training," *IJSS Ilomata International Journal of Social Science*, vol. 1, no. 3, pp. 88–102, Jul. 2020, [Online]. Available: https://www.ilomata.org/index.php/ijss

[18] M. R. S. Abdillah, R. Mayasari, and U. Enri, "Perancangan dan Implementasi Aplikasi Android Menggunakan Framework Flutter pada Penjualan Seblak Mandji," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 9, no. 1, pp. 643–654, Feb. 2025, doi: https://doi.org/10.36040/jati.v9i1.12454.

[19] Sondang, "Penerapan Metode RAD Dalam Pengembangan Sistem Informasi Pemesanan Jasa Percetakan Berbasis Web pada Percetakan Karya Sehati Jaya," *Remik: Riset dan E-Jurnal Manajemen Informatika Komputer*, vol. 8, no. 3, pp. 871–881, 2024, doi: 10.33395/remik.v8i3.13944.

[20] Hendri, Virgo, R. Komarudin, N. Afni, and Y. I. Maulana, "Application Of The Rapid Application Development (RAD) Method In Designing A Web-Based Car Rental Information System," *Journal of Information System, Informatics and Computing*, vol. 8, no. 1, pp. 73–80, Jun. 2024, doi: 10.52362/jisicom.v8i1.1515.

[21] S. Wahyuni, M. T. Raisal, and R. W. Nasution, "Qibla Direction in Various Coordinates in Indonesia using Spherical Trigonometry," *Al-Hisab: Journal of Islamic Astronomy*, vol. 1, no. 1, pp. 15–23, Mar. 2024, [Online]. Available: https://jurnal.umsu.ac.id/index.php/alhisab/article/view/17229