APLIKASI *TRACKING SALES MARKETING* BERBASIS ANDROID MENGGUNAKAN METODE *REVERSE-GEOCODING* DI PT. SRI INDAH LABETAMA

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh kelulusan
Jenjang Strata Satu (S1)
Pada Program Studi Teknik Informatika

Oleh

Lutfhi Rizaldi

361843005

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
INDONESIA MANDIRI

2022

LEMBAR PENGESAHAN

APLIKASI *TRACKING SALES MARKETING* BERBASIS ANDROID MENGGUNAKAN METODE *REVERSE-GEOCODING* DI PT. SRI INDAH LABETAMA

Oleh

Lutfhi Rizaldi

361843005

Tugas Akhir ini telah diterima dan disahkan untuk memenuhi persyaratan mencapai gelar

SARJANA TEKNIK INFORMATIKA

Pada

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER INDONESIA MANDIRI

Bandung, Juni 2022

Disahkan Oleh

Ketua Program Studi,	Dosen Pembimbing,
Chalifa Chazar, S.T, M.T	Chalifa Chazar, S.T, M.T
NIDN. 0421098704	NIDN. 0421098704

LEMBAR PERSETUJUAN REVISI

APLIKASI *TRACKING SALES MARKETING* BERBASIS ANDROID MENGGUNAKAN METODE *REVERSE-GEOCODING* DI PT. SRI INDAH LABETAMA

Oleh

Lutfhi Rizaldi

361843005

Telah melakukan sidang tugas akhir dan telah melakukan revisi sesuai dengan perubahan dan perbaikan yang diminta pada saat sidang tugas akhir.

Bandung, Juni 2022

Menyetujui

No	Nama Dosen	Keterangan	Tanda Tangan
1.	Chalifa Chazar, S.T., M.T.	Pembimbing	
2.	Moch. Ali Ramdhani, S.T., M.Kom.	Penguji 1	
3.	Yudhi W. Arthana R., S.T., M.Kom.	Penguji 2	

Mengetahui
Ketua Program Studi Teknik Informatika

Chalifa Chazar, S.T., M.T.
NIDN. 0421098704

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa :

(1) Naskah Skripsi ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik, baik di Sekolah Tinggi Manajemen Informatikan dan Komputer Indonesia Mandiri maupun perguruan tinggi lainnya.

(2) Skripsi ini murni merupakan karya penelitian saya sendiri dan tidak menjiplak karya pihak lain. Dalam hal ada bantuan atau arahan dari pihak lain maka telah saya sebutkan identitas dan jenis bantuannya di dalam lembar ucapan terima kasih

(3) Seandainya ada karya pihak lain yang ternyata memiliki kemiripan dengan karya saya ini, maka hal ini adalah di luar pengetahuan saya dan terjadi tanpa kesengajaan dari pihak saya.

Pernyataan ini saya buat dengan sesungguhnya dan apabila di kemudian hari terbukti adanya kebohongan dalam penyataan ini, maka saya bersedia menerima sanksi akademik sesuai norma yang berlaku di Sekolah Tinggi Manajemen Informatika dan Komputer Indonesia Mandiri.

Bandung, Juni 2022

Yang membuat pernyataan

Lutfhi Rizaldi

361843005

ABSTRAK

APLIKASI *TRACKING SALES MARKETING* BERBASIS ANDROID MENGGUNAKAN METODE *REVERSE-GEOCODING* DI PT. SRI INDAH LABETAMA

Oleh

Lutfhi Rizaldi

361843005

PT. Sri Indah Labetama merupakan perusahaan manufaktur yang bergerak di bidang pembuatan. Karena banyak sedikitnya order penjualan yang masuk dipengaruhi oleh tren yang terjadi di perusahaan garmen, diperlukan penambahan order penjualan disaat tren sedang turun. Banyaknya kunjungan yang dilakukan oleh *sales Marketing* merupakan upaya dalam meningkatkan nilai penjualan label, namun karena banyaknya *sales Marketing* yang kunjungan, kepala marketing dan bagian personalia mengalami kendala dalam hal monitoring aktivitas *sales Marketing* di luar kantor. Monitoring ini diperlukan karena setiap hari kepala bagian divisi marketing akan melakukan pelaporan aktivitas marketing dalam mencari order dan pelaporan itu akan ditembuskan kepada bagian personalia dan *General Manager* sebagai *daily report* agar dapat dibuktikan bahwa *sales Marketing* bersangkutan dalam kesehariannya di kantor adalah mencari order. Dari permasalahan inilah dibuat sebuah aplikasi atau sistem yang mana aplikasi atau sistem tersebut mampu memonitoring posisi *sales Marketing* ketika *sales Marketing* melakukan *visit* atau kunjungan ke *customer*. Dengan menggunakan metode *reverse-geocoding*, titik koordinat lokasi yang dikirimkan oleh *smartphone* akan dikonversi menjadi alamat yang lebih dimengerti oleh user. Hasil penelitian menunjukkan bahwa aplikasi dapat berjalan sesuai dengan yang diharapkan dan *sales Marketing* dapat termonitor oleh personalia dan kepala *sales Marketing*.

Kata Kunci : *android, java, geo-location, firestore, reverse-geocoding*

ABSTRACT

ANDROID-BASED TRACKING SALES MARKETING APPLICATION USING REVERSE-GEOCODING METHOD AT PT. SRI INDAH LABETAMA

Oleh

Lutfhi Rizaldi

361843005

PT. Sri Indah Labetama is a company engaged in the manufacture of garment accessories. Because the number of incoming sales orders is influenced by trends that occur in garment companies, it is necessary to add sales orders when the trend is down. The number of visits made by sales Marketing is an effort to increase the value of label sales, but due to the large number of sales Marketing visits, the head of the marketing department and the personnel department have problems in monitoring sales Marketing activities outside the office. This monitoring is necessary because every day the head of the marketing division will report on marketing activities in finding orders and this report will be forwarded to the personnel department and the General Manager as a daily report so that it can be proven that the sales Marketing concerned in his daily work at the office is looking for orders. From these problems, an application or system was made where the application or system was able to monitor the position of sales Marketing when sales Marketing made visits or visits to customers. By using the reverse-geocoding method, the coordinates of the location sent by the smartphone will be converted into an address that is easier for users to understand. The results show that the application can run as expected and sales Marketing can be monitored by sales Marketing personnel and heads.

Keywords: android, java, geo-location, firestore, reverse-geocoding

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT, atas limpahan rahmat dan karunia-Nya, sehingga penulis dapat menyusun laporan tugas akhir ini dengan sebaik-baiknya. Laporan yang berjudul *"Aplikasi Tracking Sales Marketing Berbasis Android Menggunakan Metode Reverse-Geocoding Di PT. Sri Indah Labetama"* ini disusun sebagai salah satu syarat untuk memperoleh kelulusan Jenjang Strata Satu (S1) pada Program Studi Teknik Informatika

Skripsi ini menjelaskan bagaimana perancangan sebuah aplikasi *tracking* untuk memonitor pergerakan atau aktivitas ketika *sales Marketing* sedang melakukan *visit* atau kunjungan ke *customer* agar dapat dimonitor secara langsung baik oleh kepala marketing masing-masing divisi maupun bagian personalia (HRD).

Penulis menyadari dalam penyusunan laporan ini masih terdapat kekurangan yang jauh dari kesempurnaan. Oleh sebab itu, penulis sangat berharap kritik dan saran yang membangun dari para pembaca agar penulis dapat memperbaiki laporan ini agar lebih sempurna.

Akhir kata, penulis berharap agar laporan *"Aplikasi Tracking Sales Marketing Berbasis Android Menggunakan Metode Reverse-Geocoding Di PT. Sri Indah Labetama"* ini dapat memberi manfaat bagi pembaca semua.

Bandung, 14 Juni 2022

Penulis

Lutfhi Rizaldi

361843005

UCAPAN TERIMA KASIH

Dalam penyusunan skripsi ini tidak terlepas dukungan dari berbagai pihak. Penulis secara khusus mengucapkan terima kasih yang sebesar besarnya kepada semua pihak yang telah membantu. Penulis banyak menerima bimbingan, petunjuk dan bantuan serta dorongan dari berbagai pihak baik yang bersifat moral maupun material. Pada kesempatan ini penulis menyampaikan rasa terima kasih yang sebesar-besarnya kepada :

1. Allah SWT dengan segala rahmat serta karunia-Nya yang memberikan kekuatan bagi penulis dalam menyelesaikan skripsi ini.
2. Kepada kedua orang tua tercinta yang selama ini telah membantu penulis dalam bentuk perhatian, kasih sayang, semangat, serta doa yang tidak hentihentinya mengalir demi kelancaran dan kesuksesan penulis dalam menyelesaikan skripsi ini.
3. Ibu Chalifa Chazar, S.T., M.T. selaku pembimbing dan Kaprodi Teknik Informatika yang telah memberikan penulis ilmu dan masukan dalam penyusunan skripsi.
4. Bapak Moch. Ali Ramdhani, S.T., M.Kom. dan Bapak Yudhi W. Arthana R., S.T., M.Kom. selaku penguji pada sidang skripsi yang telah memberikan masukan dalam skripsi.
5. PT Sri Indah Labetama yang telah memberikan kesempatan penulis untuk melakukan kegiatan penelitian dan membuat aplikasi *tracking Sales Marketing* berbasis android guna mendukung kelancaran penyusunan skripsi.

6. Bapak Yoffie Adhitya, S.T. selaku rekan sekaligus pembimbing di PT. Sri Indah Labetama yang telah meluangkan waktu, tenaga dan pikiran dalam memberikan sarannya mengenai pembuatan aplikasi *tracking Sales Marketing* berbasis android dalam mendukung kelancaran dalam penyusunan skripsi ini.
7. Bapak Dr. Chairuddin, Ir., M.M., M.T. selaku Ketua STMIK-IM Bandung yang telah memberikan ilmu kepada penulis.
8. Bapak & Ibu dosen yang telah memberikan penulis ilmu selama penulis melakukan kegiatan pembelajaran di STMIK-IM Bandung.
9. Kepala dan seluruh staff Administrasi, BAAK, BAUK dan staff Perpustakaan STMIK-IM Bandung.
10. Serta masih banyak lagi pihak-pihak yang sangat berpengaruh dalam proses penyelesaian skripsi yang yang tidak bisa penulis sebutkan satupersatu.

Semoga Allah SWT senantiasa membalas semua kebaikan yang telah diberikan. Semoga penelitian ini dapat bermanfaat bagi penulis umumnya kepada para pembaca.

Bandung, 14 Juni 2021

Lutfhi Rizaldi

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1. Latar Belakang

PT. Sri Indah Labetama merupakan perusahaan manufaktur yang bergerak di bidang pembuatan label untuk diaplikasikan sebagai aksesoris, baik untuk produk garmen maupun produk-produk yang membutuhkan label, yang mana digunakan sebagai pengenal merek dari produk tersebut maupun informasi bagi pengguna produk berupa kandungan yang terdapat di dalam produk atau cara pemakaian dan perawatan produk tersebut. Banyaknya penjualan yang terjadi menjadikan proses produksi berjalan setiap hari tanpa henti. Hal ini merupakan keuntungan bagi perusahaan karena dengan proses produksi yang berjalan setiap hari maka keberlangsungan perusahaan masih dapat dijaga dan laba dari penjualan pun semakin banyak. Namun, penjualan yang terjadi dipengaruhi pula oleh tren yang terjadi di perusahaan garmen, dimana nilai penjualan sebanding dengan permintaan yang masuk dari *buyer* kepada garment, sehingga nilai penjualan dalam satu tahun bisa mengalami kenaikan maupun penurunan. Hal ini bisa mengganggu proses produksi karena produksi bergantung kepada nilai penjualan yang terjadi.

Karena banyak sedikitnya order penjualan yang masuk dipengaruhi oleh tren yang terjadi di perusahaan garmen, diperlukan penambahan order penjualan

disaat tren sedang turun. Adapun penambahan yang dimaksud adalah penambahan jumlah order penjualan yang masuk dari *new customer* sebagai pengisi kekurangan order penjualan yang biasa diisi oleh *customer existing* sehingga nilai penjualan yang terjadi pada saat tren sedang turun bernilai tetap dan produksi tetap berjalan. Penambahan jumlah order ini biasa dilakukan oleh *Sales Marketing* dengan cara melakukan kunjungan rutin kepada *customer existing* untuk dilakukan *follow-up* order maupun *develope sample* baru. Selain itu, kunjungan juga dilakukan terhadap *new customer* yang baru melakukan *develope sample* agar sample segera turun order.

Banyaknya kunjungan yang dilakukan oleh *Sales Marketing* merupakan upaya dalam meningkatkan nilai penjualan label, namun karena banyaknya *Sales Marketing* yang kunjungan, kepala marketing dan bagian personalia mengalami kendala dalam hal monitoring aktivitas *Sales Marketing* di luar kantor. Monitoring ini diperlukan karena setiap hari kepala bagian divisi marketing akan melakukan pelaporan aktivitas marketing dalam mencari order dan pelaporan itu akan ditembuskan kepada bagian personalia dan *General Manager* sebagai *daily report* agar dapat dibuktikan bahwa *Sales Marketing* bersangkutan dalam kesehariannya di kantor adalah mencari order.

Adapun sistem yang sudah berjalan dalam memonitoring kegiatan marketing ketika *visit* adalah dengan dibuatnya surat tugas kunjungan yang dibuat oleh *Sales Marketing* sebagai bukti bahwa *Sales Marketing* akan melakukan kunjungan ke *customer* besok. Adapun surat tugas tersebut berisikan nama *Sales Marketing* serta tempat tujuan kunjungan dengan apa saja yang akan dilakukan

oleh marketing selama kunjungan. Namun, hal ini masih dirasa menjadi masalah karena tidak dapat mengetahui secara pasti apakah benar *Sales Marketing* pada saat jam kerja sedang berada di tempat *customer* atau tidak.

Dari permasalahan tersebut, diperlukan usulan sebuah aplikasi atau sistem yang mana aplikasi atau sistem tersebut mampu memonitoring posisi *Sales Marketing* ketika *Sales Marketing* melakukan *visit* atau kunjungan ke *customer*. Dengan adanya aplikasi atau sistem yang mampu memonitoring posisi *Sales Marketing* ketika sedang melakukan *visit* atau kunjungan ke *customer* dapat memudahkan bagi kepala bagian divisi marketing maupun bagian personalia (HRD) dalam memonitor aktivitas *Sales Marketing* selama berada diluar kantor. Kepala bagian divisi marketing cukup mencocokkan antara surat tugas kunjungan dengan posisi *Sales Marketing* terakhir di *update*.

Dalam penelitian ini, diusulkan pula perancangan aplikasi *tracking* berbasis android dengan menggunakan metode *reverse-geocoding*, dimana metode ini mengubah alamat posisi GPS yang didapat menjadi nama lokasi yang sebenarnya sehingga didapatkan posisi yang dapat lebih dimengerti oleh *user* ketimbang posisi koordinat GPS yang hanya berupa koordinat *Latitude* dan *Longitude*. Selain itu, penggunaan metode ini karena nantinya aplikasi yang digunakan oleh *Sales Marketing* akan melaporkan titik koordinat ke *server* dan aplikasi yang digunakan oleh kepala bagian divisi marketing akan mengambil titik koordinat dari *server* dan mengubah titik koordinat tersebut menjadi alamat yang lebih nyata. Basis Android dipilih karena perangkat yang paling banyak digunakan adalah *handphone* dengan sistem operasi Android.

Adapun tujuan dilakukan penelitian ini dengan diajukannya perancangan aplikasi *tracking* atau monitoring berbasis android bertujuan agar atasan bisa memonitoring secara langsung (*tracking*) setiap jadwal kunjungan yang diberikan oleh perusahaan benar-benar dapat dilaksanakan oleh *team marketing*. Jika dari jadwal yang diberikan terdapat ketidaksesuaian maka produktifitas dari *marketing* dinilai berkurang karena akan menimbulkan efek terhadap peluang bertambahnya *customer* baru maupun order dari *customer existing*. Dengan begitu, memudahkan masing-masing pihak dalam melakukan *cross-check* kebenaran antara surat tugas kunjungan dengan aktual posisi *Sales Marketing* di lapangan.

1.2. Identifikasi Masalah

Berdasarkan latar belakang diatas, maka masalah yang ada di dalam latar belakang diatas dapat di identifikasikan adalah bagaimana kepala divisi marketing memonitoring secara langsung posisi *Sales Marketing* ketika sedang melakukan kunjungan ke *customer* ?

1.3. Tujuan Penelitian

Tujuan dari penelitian ini adalah membuat aplikasi berbasis android untuk *Sales Marketing* yang dapat mengirimkan posisi koordinat saat itu juga dan mengirimkannya ke *server* sehingga aplikasi yang digunakan oleh kepala divisi

marketing dapat mengambil titik koordinat yang dikirimkan oleh *Sales Marketing* agar didapatkan alamat nyata atau posisi aktual dari *Sales Marketing*.

1.4. Batasan Masalah

Adapun dalam penelitian ini perlu dibatasi pembahasannya agar arah pembahasan tidak melebar ke mana-mana. Adapun poin-poin pembahasannya adalah sebagai berikut :

1. GPS digunakan untuk mendapatkan titik koordinat.
2. Penggunaan metode *Reverse-Geocoding* sebagai metode dasar dalam menentukan posisi aktual dari aplikasi.
3. Perancangan aplikasi menggunakan Bahasa pemrograman Java.
4. Basis data yang digunakan adalah Firebase
5. Perancangan aplikasi dilakukan di PT. Sri Indah Labetama
6. Aplikasi yang dibuat hanya satu, yang membedakannya adalah akses ketika login.

1.5. Metode Penelitian

Dalam penelitian ini menggunakan metode penelitian sebagai berikut :

1.5.1. Identifikasi Masalah

Mengidentifikasi permasalahan yang sedang terjadi di PT. Sri Indah Labetama, dimana kepala bagian divisi marketing dan bagian personalia kesulitan untuk memantau dan memonitoring secara langsung *Sales Marketing* yang sedang melakukan *visit* ke *customer*, sehingga diusulkan sebuah rancangan aplikasi *tracking* agar kepala bagian divisi *Sales Marketing* dapat memantau dan memonitoring secara langsung posisi dari *Sales Marketing*nya.

1.5.2. Teknik Pengumpulan Data

Data yang diperoleh selama melakukan penelitian yang dilakukan di PT. Sri Indah Labetama merupakan data-data berupa nama-nama *Sales Marketing*. Adapun metodologi pengumpulan datanya adalah sebagai berikut :

1. Wawancara

   Wawancara dilakukan selama penelitian berlangsung, dengan narasumber yang berhubungan dengan tema penelitian.

2. Studi Pustaka

   Untuk memperkuat dasar teoritis dari penelitian, selama penelitian berlangsung ditunjang dengan studi literatur dari

buku-buku dan sumber informasi dari internet yang berhubungan dengan tema penelitian.

1.5.3. Metode Perancangan

Metode perancangan perangkat lunak yang dipakai adalah metode *Prototype*. *Prototyping* merupakan metode pengembangan perangat lunak, yang berupa model fisik kerja sistem dan berfungsi sebagai versi awal dari sistem. Dengan metode *prototyping* ini akan dihasilkan *prototype* sistem sebagai perantara pengembang dan pengguna agar dapat berinteraksi dalam proses kegiatan pengembangan sistem informasi (Purnomo, 2017). Agar proses pembuatan *prototype* ini berhasil dengan baik adalah dengan mendefinisikan aturan-aturan pada tahap awal, yaitu pengembang dan penguna harus satu pemahaman bahwa *prototype* dibangun untuk mendefinisikan kebutuhan awal. *Prototype* akan dihilangkan atau ditambahkan pada bagiannya sehingga sesuai dengan perencanaan dan analisis yang dilakukan oleh pengembang sampai dengan ujicoba dilakukan secara simultan seiiring dengan proses pengembangan. Tahapan-tahapan dalam *Prototyping* adalah sebagai berikut:

1. Pengumpulan kebutuhan

   Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

2. Membangun *prototyping*

   Membangun prototyping dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat *input* dan format *output*).

3. Evaluasi *protoptyping*

   Evaluasi ini dilakukan oleh pelanggan apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah selanjutnya akan diambil. Jika tidak prototyping direvisi dengan mengulang langkah sebelumnya dari awal.

4. Mengkodekan sistem

   Dalam tahap ini prototyping yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.

5. Menguji sistem

   Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan *White Box, Black Box, Basis Path*, pengujian arsitektur dan lain-lain.

6. Evaluasi Sistem

   Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan. Jika ya, langkah selanjutnya akan dilakukan dan jika tidak, akan diulangi lagi dari mulai mengkodekan sistem .

7. Menggunakan sistem

Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan.

1.6. Sistematika Penulisan

Adapun dalam penulisan skripsi ini, penulis menggunakan sistematika penulisan sebagai berikut :

Bab I : Pendahuluan

Pada bab ini akan dijelaskan mengenai latar belakang, identifikasi masalah, tujuan penelitian, batasan masalah, dan metode penelitian yang digunakan dalam penelitian ini.

Bab II : Landasan Teori

Pada bab ini akan dijelaskan tentang landasan teoritis mengenai pemecahan masalah dan teori-teori yang berkaitan terhadap penelitian .

Bab III : Analisa masalah dan perancangan program

Pada bab ini akan dijelaskan menjelaskan tentang analisis masalah yang terjadi di PT. Sri Indah Labetama serta analisis berupa perancangan aplikasi *tracking* berbasis Android

Bab IV : Implementasi dan Uji Coba

Pada bab ini akan dijelaskan mengenai implementasi atas perancangan aplikasi *tracking* serta uji coba berupa pengukuran keakuratan lokasi atau titik koordinat.

Bab V: Penutup

Pada bab ini, penulis memaparkan kesimpulan dan saran untuk pengembangan aplikasi di masa yang akan datang

BAB II

LANDASAN TEORI

2.1. Pengertian *Prototype*

Prototyping merupakan metode pengembangan perangat lunak, yang berupa model fisik kerja sistem dan berfungsi sebagai versi awal dari sistem (Purnomo, 2017). Tujuannya adalah mengembangkan model menjadi sistem final. Artinya sistem akan dikembangkan lebih cepat dari pada metode tradisional dan biayanya menjadi lebih rendah. Ciri khas dari metodologi ini adalah pengembang sistem (*system developer*), klien, dan pengguna dapat melihat dan melakukan eksperimen dengan bagian dari sistem komputer dari sejak awal proses pengembangan.

Dengan *prototype* yang terbuka, model sebuah sistem (atau bagiannya) dikembangkan secara cepat dan dipoles dalam diskusi yang berkali-kali dengan klien. Model tersebut menunjukkan kepada klien apa yang akan dilakukan oleh sistem, namun tidak didukung oleh rancangan desain struktur yang mendetil. Pada saat perancang dan klien melakukan percobaan dengan berbagai ide pada suatu model dan setuju dengan desain final, rancangan yang sesungguhnya dibuat tepat seperti model dengan kualitas yang lebih bagus.

Protoyping membantu dalam menemukan kebutuhan di tahap awal pengembangan,terutama jika klien tidak yakin dimana masalah berasal. Selain itu protoyping juga berguna sebagai alat untuk mendesain dan memperbaiki *user*

interface – bagaimana sistem akan terlihat oleh orang-orang yang menggunakannya. Tahapan-tahapan dalam *Prototyping* adalah sebagai berikut:

1. Pengumpulan kebutuhan

   Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

2. Membangun *prototyping*

   Membangun prototyping dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat *input* dan format *output*).

3. Evaluasi *protoptyping*

   Evaluasi ini dilakukan oleh pelanggan apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah selanjutnya akan diambil. Jika tidak prototyping direvisi dengan mengulang langkah sebelumnya dari awal.

4. Mengkodekan sistem

   Dalam tahap ini prototyping yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.

5. Menguji sistem

   Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini

dilakukan dengan *White Box, Black Box, Basis Path*, pengujian arsitektur dan lain-lain.

6. Evaluasi Sistem

   Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan. Jika ya, langkah selanjutnya akan dilakukan dan jika tidak, akan diulangi lagi dari mulai mengkodekan sistem .

7. Menggunakan sistem

   Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan.

2.2. Pengertian Android

Android adalah suatu sistem operasi pada *smartphone* atau *tablet* yang mempunyi banyak fitur didalamnya untuk mempermudah kehidupan manusia dan sampai sekarang terus berkembang semakin canggih (Galih Pradana & Nita, 2019).

Android merupakan OS (Sistem Operasi) *Mobile* yang tumbuh ditengah OS lainnya yang berkembang dewasa ini. OS lainnya seperti *Windows Mobile*, OS i-*Phone, Symbian*, dan masih banyak lagi selain itu. Akan tetapi, OS yang ada ini menjalankannya dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat dari potensi yang cukup besar dari aplikasi pihak ketiga.

Dengan menyediakan sebuah platform pengembangan yang terbuka, pengembang Android menawarkan kemampuan untuk membangun aplikasi yang sangat kaya dan inovatif. Pengembang bebas untuk mengambil keuntungan dari perangkat keras, akses informasi lokasi, menjalankan background services, mengatur alarm, tambahkan pemberitahuan ke status bar, dan banyak lagi. Android bergantung pada versi Linux 2.6 untuk layanan sistem inti seperti keamanan, manajemen memori, manajemen proses, network stack, dan model driver. Kernel juga bertindak sebagai lapisan abstraksi antara hardware dan seluruh software stack. Sejak saat itu, Android terus mengembangkan sistem operasinya untuk digunakan pada perangkat mobile. Berikut urutan versi Android dari awal hingga terbaru:

1. Android 1.0 Alpha

   Versi Android satu ini telah dilengkapi fitur dukungan akses web browser, streaming youtube, pemutar media, google map, serta sinkronisasi dengan aplikasi google lainnya walaupun belum dirilis secara komersial.

2. Android 1.1 Beta

   Versi beta ini juga belum dirilis secara komersial dan hanya diperuntukkan untuk satu perangkat. Dalam versi ini Android meningkatkan beberapa fitur seperti memperbaiki bugs, rincian lokasi pada google maps dan fitur menyembunyikan serta menampilkan tombol panggilan.

3. Android 1.5 Cupcake

   2 bulan kemudian setelah dirilis versi beta, Android mengeluarkan versi Android 1.5 Cupcake, tepatnya tanggal 30 April 2009. Dimana versi ini untuk pertama kali diperkenalkan secara komersial. Nah, dari sinilah Android mulai memakai nama makanan manis untuk menamai versi Android yang dirilisnya. Fitur tambahan dari versi ini adalah layer otomatis, widget serta keyboard virtual.

4. Android 1.6 Donut

   Urutan versi Android selanjutnya ada Android 1.6 Donut yang dirilis pada tanggal 15 september 2009. Fitur yang ditambahkan pada versi ini adalah persentase daya baterai, fasilitas pencarian android market atau play store dan dukungan gestur.

5. Android 2.0 Eclair

   Versi Android satu ini resmi memperkaya fiturnya pada tanggal 26 Oktober 2009. Versi 2.0 ini diberi nama éclair dan menambah fitur multi touch, live wallpaper, perubahan tampilan antarmuka dan dukungan browser untuk HTML5.

6. Android 2.2 Froyo

   Kali ini Android meluncurkan versi terbaru Android 2.2 Froyo pada tanggal 20 Mei 2010. Pada versi ini Android semakin dikenal luas oleh vendor atau pabrikan ponsel. Fitur unggulan pada versi ini adalah peningkatan fitur USB tethering dan hotspot WIFI,

memperbesar gambar pada galeri dengan gestur, serta dukungan animasi GIF pada web browser.

7. Android 2.3 Gingerbread

   Fitur ini merupakan urutan versi Android yang ke tujuh. Saat diluncurkan versi ini Android telah menjadi sistem mobile yang populer di dunia. Fitur ini dirilis pertama kali pada tanggal 6 desember 2010 dengan tambahan fitur copy paste dengan memilih kata melalui layer yang ditekan serta dukungan beberapa sensor lainnya.

8. Android 3.0 Honeycomb

   Diluncurkan pertama kali pada tanggal 22 februari 2011. Namun fitur ini hanya dikhususkan untuk perangkat tablet. Fitur yang dimiliki antara lain dukungan obrolan video dengan Google Talk, dukungan prosesor *multi core* dan percepatan saat berpindah aplikasi yang sedang berjalan dengan fitur multitasking *recent apps*.

9. Android 4.0 Ice Cream Sandwich

   Fitur ini memiliki tambahan honeycomb yang bisa berjalan pada *smartphone* yang sebelumnya hanya ditujukan untuk tablet PC. Selain itu terdapat fitur perbaikan antarmuka dan kostumisasi widget.

10. Android 4.1 Jelly Bean

    Versi ini meningkatkan performa tampilan antarmuka. Widget yang dapat disesuaikan ukuran dan diatur sendiri serta UI yang semakin smooth merupakan fitur terbarunya. Terdapat pula keyboard yang bisa dikostumisasi oleh pengguna dan dukungan gestur pada keyboard.

11. Android 4.4 KitKat

    Fitur yang dibawa versi ini adalah dukungan sensor batching, pengoptimalan kinerja terhadap perangkat dengan spesifikasi rendah. Terdapat pula WebViews yang berbasiskan Chromium dan step detector.

12. Android 5.0 Lollipop

    Versi yang dikembangkan Android ini tak hanya menjadi sistem operasi pada perangkat *smartphone*. Namun bisa juga berjalan pada perangkat mobile lainnya seperti Android TV dan google fit. Rilis pada tanggal 25 juni 2014, versi ini membawa fitur baru seperti user interface dan fitur factory reset protection.

13. Android 6.0 Marshmallow

    Versi ini memiliki fitur tambahan yakni sensor sidik jari untuk mengakses *smartphone*. Fasilitas menjalankan beberapa aplikasi pada tata letak layer dengan dukungan multi window, dukungan platform virtual reality. Ada Pula kemampuan pada mode data saver untuk mengurangi pemakaian bandwidth.

14. Android 7.0 Nougat

    Fokus pada urutan versi Android ke 14 ini yakni meningkatkan performa user interface. Hal ini membuat Android lebih intuitif dan penggunaan aplikasi secara bersamaan lebih banyak pada fitur multi window. Fitur tambahan lain yakni mode malam, keyboard default dan dukungan panggilan multi end point.

15. Android 8.0 Oreo

    Fitur tambahan versi ini adalah Autofill yang memberikan kemudahan dalam mengisi formulir. Selain itu user interface pada versi ini lebih simpel agar lebih mudah dalam mengakses aplikasi.

16. Android 9.0 Pie

    Versi ini memiliki fitur unggulan kemampuan AI atau kecerdasan buatan. Dimana fitur ini mampu menganalisa dan mempelajari pola pemakaian secara otomatis. Selain itu ada *adaptive brightness* dan *bezel less.*

17. Android 10

    Urutan versi Android ini tak diberi nama seperti pendahulunya dengan menggunakan nama makanan manis. Penamaan Android 10 merupakan tanda petunjuk yang digunakan untuk menunjukkan bahwa Android telah melewati 1 dekade. Versi ini lebih berfokus pada penyempurnaan mode malam dan peningkatan fitur *sound amplifier*.

18. Android 11

    Android 11 punya fitur Balon yang bisa membuat Anda melanjutkan percakapan setelah mengakses aplikasi lain. Akses chat kapan saja dan di mana saja lebih mudah. Android 11 juga punya fitur perekam layar (*Screen Recording*) bawaan. Ada juga fitur izin satu kali ke apliaksi yang meminta akses.

19. Android 12

    Android versi 12 ini dirilis perdana pada 4 Oktober 2021. Urutan Android versi terbaru ini memiliki update besar pada Desain Material yang kemudian disebut sebagai “Material You.”

    Sistem operasi bisa secara otomatis menghasilkan tema warna untuk menu sistem dan aplikasi yang didukung menggunakan warna wallpaper pengguna. Android 12 juga dilengkapi fitur untuk ambil screenshot tampilan web secara utuh atau bergulir. Pengguna juga bisa mencegah aplikasi untuk menggunakan aplikasi dan mikrofon melalui pengaturan cepat.

20. Android 13

    Android 13 adalah versi sistem operasi Android ke-20. OS Android 13 pertama kali diperkenalkan pada 10 Februari 2022. Urutan Android terbaru ini dirilis sekitar 4 bulan setelah versi stabil Android 12 dirilis. Android 13 fokus pada penyempurnaan fitur pada Android 12L, meningkatkan privasi, keamanan, dan optimalisasi UI.

Android 13 punya fitur pemilih foto baru. Fitur ini memungkinkan Anda bisa atur foto atau video tertentu yang bisa dibagikan dengan aplikasi. Berbeda seperti versi sebelumnya yang memberi akses ke semua foto dan video di *library*. Di Android 13 pengguna juga bisa kustomisasi ikon aplikasi dengan warna yang sama seperti tema atau wallpaper.

2.3. Pengertian Java

Java merupakan bahasa pemrograman yang disusun oleh James Gosling yang dibantu oleh rekan-rekannya di suatu perusahaan perangkat lunak yang bernama Sun Microsystems, pada tahun 1991. Bahasa pemrograman ini mula-mula diinisialisasi dengan nama "Oak", namun pada tahun 1995 diganti namanya menjadi "Java".

Java adalah bahasa pemrograman yang multi-device. Sekali anda menuliskan sebuah program dengan menggunakan Java, anda dapat menjalankannya hampir di semua komputer dan perangkat lain yang support Java, dengan sedikit perubahan atau tanpa perubahan sama sekali dalam kodenya (Afrizal Subhan, 2017).

Adapun kelebihan dari Bahasa pemrograman Java adalah sebagai berikut :

1. Mudah digunakan

   Dasar dari Java adalah bahasa pemrograman C++. Meskipun bahasa pemrograman tersebut cukup kuat, tetapi tergolong konteks

dan tidak cukup untuk berbagai kebutuhan Java. Java dibangun dari dan menjadi semacam peningkatan dari bahasa pemrograman tersebut. Hal ini membuat Java menjadi bahasa pemrograman yang kuat dan sederhana untuk digunakan.

2. Berorientasi pada objek

   Ini terkait dengan sifatnya yang merupakan bahasa pemrograman berorientasi objek. Hal ini berbeda dengan bahasa pemrograman C++ yang bisa dianggap semi beriorentasi pada objek. Java memiliki beberapa fitur dari bahasa pemrograman berorientasi objek atau *Object-Oriented Programming Language* (OOP). Hal-hal tersebut di antaranya adalah *abstraction*, *encapsulation*, *inheritance* dan *polymorphism*.

3. Keamanan

   Pada awalnya, Java ditujukan untuk perangkat mobile yang bertukar data lewat jaringan. Hal ini membuat bahasa Java dibangun dengan tingkat keamanan tinggi.

4. Bisa digunakan di berbagai platform

   Sebuah program idealnya bisa bekerja terlepas dari platform apa yang digunakan untuk mengeksekusinya. Java ditulis sebagai bahasa pemrograman yang portabel dan bisa digunakan lintas platform. Hal tersebut membuatnya dapat digunakan di berbagai sistem operasi, hardware, ataupun perangkat.

Ada tiga komponen penting dari Java. Ketiga komponen tersebut adalah:

1. JDK

   Java Development Kit (JDK) merupakan komponen inti dari Java. Komponen ini memberikan semua *tools, executables, binaries* yang diperlukan untuk menyusun, men-*debug*, dan mengeksekusi sebuah program Java.

2. JVM

   Java Virtual Machine (JVM) kerap dianggap sebagai jantung dari bahasa pemrograman Java. Ketika menjalankan program Java, JVM bertugas untuk mengonversi *byte code* menjadi kode yang lebih spesifik.

3. JRE

   Java Runtime Environment (JRE) merupakan implementasi dari JVM. JVM memberikan platform untuk mengeksekusi program-program Java.

2.4. Pengertian Firebase

Firebase adalah suatu layanan dari Google untuk memberikan kemudahan bahkan mempermudah para *developer* aplikasi dalam mengembangkan aplikasinya. Firebase alias BaaS (*Backend as a Service*) merupakan solusi yang ditawarkan oleh Google untuk mempercepat pekerjaan *developer*.

Firebase didirikan pertama kali pada tahun 2011 oleh Andrew Lee dan James Tamplin. Produk Firebase yang pertama kali adalah *Realtime Database*. *Realtime Database* digunakan *developer* untuk menyimpan data dan *synchronize* ke banyak *user*. Kemudian ia berkembang sebagai layanan pengembang aplikasi. Pada bulan Oktober 2014, perusahaan tersebut diakuisisi oleh Google.

Layanan-layanan yang tersedia dari Firebase ada 2 pilihan, di antaranya:

1. SPARK : Layanan secara gratis.
2. BLAZE : Dikenakan biaya sesuai dengan pemakaian layanan

Fitur-fitur yang terdapat di dalam Firebase adalah sebagai berikut :

1. Firebase Analytics

   Fitur *Analytics* merupakan fitur yang dapat digunakan untuk mengumpulkan data dan pelaporan pada aplikasi Android maupun iOS. *Developer* dapat melihat bagian mana saja dari aplikasi yang menjadi sering untuk digunakan oleh user.

2. Firebase Cloud Messaging and Notifications

   FCM (*Firebase Cloud Messaging*) adalah fitur yang menyediakan koneksi yang tangguh dan hemat baterai antar server ataupun device. *Developer* bisa mengirim dan menerima pesan pada notifikasi Android, iOS dan web tanpa dikenakan biaya. Pesan notifikasi ini juga terintegrasi dengan Google Analytics for Firebase. *Developer* dapat memantau suatu efektivitas dari satu

dashboard tanpa menggunakan *coding* atau membuat program sendiri.

3. Firebase Authentication

   Firebase Authentication adalah salah satu layanan *back-end*, fitur Android dan iOS, SDK yang dapat digunakan dengan mudah dan juga memiliki tampilan interface yang siap digunakan dalam mengidentifikasi pengguna ke aplikasi yang telah diciptakan. Firebase Authentication mendukung autentifikasi menggunakan nomor telepon, sandi, dan juga penyedia identitas gabungan populer seperti Google, Facebook, dan sebagainya.

   Firebase Authentication ini juga terintegrasi dengan layanan Firebase lainnya. Sistem ini menggunakan berbagai jenis standar industri seperti OAuth 2.0 dan OpenID Connect, yang memudahkan integrasi dengan backend khusus.

4. Firebase Cloud Firestore

   Cloud Firestore adalah database yang memiliki sifat fleksibel dan terukur dalam pengembangan perangkat contohnya seluler, web, dan server di dalam Firebase dan Google Cloud Platform. Cloud *Firestore* juga dapat membuat data Anda tetap terkoneksi pada aplikasi *user* melalui *listener realtime* dan menawarkan layanan secara *offline* untuk *mobile apps* dan *web*. Cloud Firestore juga merupakan database NoSQL dan juga dapat diakses melalui SDK *real* oleh iOS apps, Android, dan Web.

5. Firebase Realtime Database

   Firebase Realtime Database adalah database yang di-host pada cloud. Data tersebut disimpan dan dieksekusi berbentuk JSON kemudian disinkronkan secara realtime kepada setiap user yang terhubung. Kemampuan lain dari Firebase Realtime Database ialah dapat responsif bahkan saat *offline* karena SDK Firebase Realtime Database dapat menyimpan data langsung ke *disk device* atau memori lokal. Setelah perangkat terhubung kembali dengan internet, perangkat *user* dapat menerima setiap *update* yang terjadi.

6. Firebase Hosting

   Firebase Hosting adalah layanan *hosting* pada konten *website* dengan satu instruksi yang diterapkan pada *website* dan menampilkan konten statis ataupun dinamis melalui CDN (jaringan penayangan konten) secara jangkuan global dan cepat. Kegunaan dari fitur ini adalah dapat menampilkan konten melalui koneksi yang aman, mengirimkan konten secara cepat serta mendukun semua jenis konten untuk di simpan di hosting mulai dari file HTML dan CSS hingga API atau layanan mikro Express.js

2.5. Pengertian NoSQL

Basis data NoSQL tidak menggunakan model data rasional, dapat menyimpan data dalam ukuran besar dan juga memperbolehkan data untuk disimpan didalam record yang tidak mempunyai skema yang sudah tertentu (Suliyanti, 2019).

Tujuan utama dari penggunaan database NoSQL adalah untuk penyimpanan data yang terdistribusi dengan kebutuhan penyimpanan data yang besar. Umumnya NoSQL digunakan untuk penggunaan big data dan aplikasi web yang *real-time*. Adapun kelebihan dari database NoSQL adalah sebagai berikut :

1. Performa

   Database NoSQL disebut memiliki performa yang lebih unggul jika dibandingkan dengan database SQL. Hal itu disebabkan semua informasi terdapat di dalam satu database saja. Sementara itu, dengan SQL maka pengguna harus mengkueri data di beberapa tabel terlebih dahulu. Namun, dengan NoSQL semua berada di dalam satu tabel sehingga mengambil data bisa dilakukan dengan cepat. Beberapa bentuk database NoSQL bahkan dapat melakukan hingga sepuluh ribu kueri per detik.

2. Skalabilitas

   Kelebihan selanjutnya dari NoSQL adalah dari skalabilitasnya. *Database* ini menggunakan penskalaan *horizontal* dengan membagi data dan menempatkannya di beberapa mesin. Sementara itu, penskalaan

vertikal berarti menambahkan lebih banyak sumber daya ke mesin. Sehingga bisa disebut lebih mahal dan membutuhkan banyak sumber daya. Penskalaan *vertikal* juga tidak mudah diterapkan seperti penskalaan *horizontal*. Jadi, bisa dibilang bahwa penggunaan NoSQL bisa menjadi lebih mudah dan murah dibandingkan dengan SQL.

3. Fleksibilitas

   NoSQL lebih fleksibel sehingga jauh lebih mudah untuk menguji ide dan melakukan pembaruan. Hal tersebut sangat penting dalam aplikasi modern di mana sering terjadi perubahan struktur data yang harus cepat dan mudah.

Adapun kekurangan dari database NoSQL adalah sebagai berikut :

1. Membutuhkan banyak database

   Penggunaan NoSQL sangat terspesialisasi untuk penggunaan tertentu. Hal tersebut berbeda dengan SQL yang lebih umum dan bisa memenuhi berbagai kebutuhan. Jadi, dengan NoSQL maka diperlukan beberapa jenis database dan model data untuk menggunakannya. Bahkan, mungkin masih diperlukan penggunaan beberapa bentuk SQL untuk membantu mempersingkat prosesnya.

2. Ukuran database bisa menjadi sangat besar

   NoSQL tidak dirancang untuk menghapus duplikasi data sehingga ukuran database bisa menjadi sangat besar. Hal itu membutuhkan lebih banyak tempat penyimpanan data untuk dipersiapkan jika ingin menggunakan NoSQL.

3. Pengelolaan yang tidak mudah

   Mengelola data dalam jumlah sangat besar bukanlah hal yang mudah. Itulah mengapa pengelolaan data di NoSQL menjadi lebih kompleks. Meskipun tujuan menggunakan NoSQL adalah untuk mengelola data dalam jumlah besar menjadi lebih sesederhana mungkin, tapi hal tersebut bukanlah hal yang mudah. Karena itu, dalam mengelola hal yang satu ini dibutuhkan lebih banyak usaha ekstra karena memang sukup sulit.

Terdapat 4 tipe utama dari *database* NoSQL sebagai berikut :

1. *Document database*

   Dalam *document database* menyimpan data dalam dokumen yang mirip dengan objek JSON (*JavaScript Object Notation*). Konsep dari *document database* ini lebih efisien dan fleksibel. Program akan lebih mudah dikembangkan karena *document database* akan menyesuaikan penyimpanan data berdasarkan kebutuhan program. Jenis database ini sangat cocok digunakan untuk database yang bertujuan umum. Selain itu, *document database* juga mampu mengakomodasi *volume* data yang besar.

2. *Key-value database*

   Jenis database ini lebih sederhana karena setiap item berisi *key* dan *value* sebagai tempat akses data. Sebuah *value* atau nilai biasanya hanya diambil dengan mereferensikan dari *key* atau kuncinya. Jadi, mempelajari cara membuat kueri untuk *key-value* tertentu bisa lebih

sederhana. *Key-value database* ini lebih sesuai untuk penyimpanan data dalam jumlah besar yang tidak perlu kueri yang rumit untuk mengambilnya.

3. *Column-based database*

   Column-based database memberikan banyak fleksibilitas daripada *database* yang relasional karena setiap baris tidak diharuskan memiliki kolom yang sama. Setiap kolom dibuat secara terpisah dan nilai dalam database kolom tunggal disimpan secara berdekatan. Jenis *database* ini memberikan kinerja tinggi pada *aggregation queries* seperti SUM, *Count*, AVG, hingga MIN karena datanya sudah tersedia di kolom. *Column-based* ini banyak digunakan untuk mengelola data *warehouse, business intelligence*, hingga CRM.

4. *Graph database*

   Graph database menyimpan data dalam *node* dan *edge*. *Node* biasanya menyimpan informasi tentang orang, tempat, dan benda-benda. Sementara itu, *edge* menyimpan informasi tentang hubungan antar *node*. Jenis database yang satu ini lebih unggul dalam penggunaan di mana pengguna perlu mencari tahu hubungan atau pola. Misalnya untuk social network, deteksi penipuan, logistik hingga rekomendasi.

2.6. Pengertian GPS

GPS adalah singkatan dari *Global Positioning System*, yang merupakan sistem navigasi dengan menggunakan teknologi satelit yang dapat menerima sinyal dari satelit (Alfeno & Devi, 2017). GPS (*Global Positioning System*) adalah sistem satelit navigasi dan penentuan posisi yang dimiliki dan dikelola oleh Amerika Serikat. Sistem ini didesain untuk memberikan posisi dan kecepatan tiga-dimensi serta informasi mengenai waktu, secara kontinyu di seluruh dunia tanpa bergantung waktu dan cuaca, kepada banyak orang secara simultan. Pada saat ini, sistem GPS sudah banyak digunakan orang di seluruh dunia. Di Indonesia pun, GPS sudah banyak diaplikasikan terutama yang terkait dengan aplikasi-aplikasi yang menuntut informasi tentang posisi. Dibandingkan dengan sistem dan metode penentuan posisi lainnya, GPS mempunyai banyak kelebihan dan menawarkan lebih banyak keuntungan, baik dalam segi operasionalisasinya maupun kualitas posisi yang diberikan. Posisi suatu titik biasanya dinyatakan dengan koordinat (dua-dimensi atau tiga-dimensi) yang mengacu pada suatu sistem koordinat tertentu. Sistem koordinat itu sendiri didefinisikan dengan menspesifikasi tiga parameter berikut, yaitu:

- Lokasi titik nol dari sistem koordinat
- Orientasi dari sumbu-sumbu koordinat, dan
- Besaran (*kartesian, Curvilinear*) yang digunakan untuk mendefinisikan posisi suatu titik dalam sistem koordinat tersebut.

Setiap parameter dan sistem koordinat tersebut dapat dispesifikasikan lebih lanjut, dan bergantung pada spesifikasi parameter yang digunakan maka dikenal beberapa jenis sistem koordinat.

2.7. Pengertian *Reverse Geocoding*

Merupakan API yang digunakan untuk mengonversi alamat ke sebuah posisi *latitude* atau *longitude* sehingga ditempatkan secara akurat pada peta. *Reverse Geocoding* adalah proses kebalikan dari *Geocoding* yaitu untuk mendapat lokasi dari latitude atau longitude. Informasi yang didapatkan berupa koordinat, lokasi alamat dan jarak yang berarah dari titik referensi (Fauzi, 2021).

2.8. Pengertian *Flowchart*

Flowchart atau sering disebut dengan diagram alir merupakan suatu jenis diagram yang merepresentasikan algoritma atau langkah-langkah instruksi yang berurutan dalam sistem. seorang analis sistem menggunakan *flowchart* sebagai bukti dokumentasi untuk menjelaskan gambaran logis sebuah sistem yang akan dibangun kepada *programmer*. Dengan begitu, *flowchart* dapat membantu untuk memberikan solusi terhadap masalah yang bisa saja terjadi dalam membangun sistem. Pada dasarnya, *flowchart* digambarkan dengan menggunakan simbol-simbol. Setiap simbol mewakili suatu proses tertentu. Sedangkan untuk

menghubungkan satu proses ke proses selanjutnya digambarkan dengan menggunakan garis penghubung (Rosaly & Prasetyo, 2019).

Fungsi utama dari *flowchart* adalah memberi gambaran jalannya sebuah program dari satu proses ke proses lainnya. Sehingga, alur program menjadi mudah dipahami oleh semua orang. Selain itu, fungsi lain dari *flowchart* adalah untuk menyederhanakan rangkaian prosedur agar memudahkan pemahaman terhadap informasi tersebut. *Flowchart* sendiri terdiri dari lima jenis, masing-masing jenis memiliki karakteristik dalam penggunaanya. Berikut adalah jenis-jenisnya:

1. *Flowchart* Dokumen

   Pertama ada *flowchart* dokumen (*document flowchart*) atau bisa juga disebut dengan *paperwork flowchart*. *Flowchart* dokumen berfungsi untuk menelusuri alur form dari satu bagian ke bagian yang lain, termasuk bagaimana laporan diproses, dicatat, dan disimpan.

2. *Flowchart* Program

   Selanjutnya kita akan membahas *flowchart* program. *Flowchart* ini menggambarkan secara rinci prosedur dari proses program. *Flowchart* program terdiri dari dua macam, antara lain: *flowchart* logika program (*program logic flowchart*) dan *flowchart* program komputer terinci (*detailed computer program flowchart*).

3. *Flowchart* Proses

   Flowchart proses adalah cara penggambaran rekayasa industrial dengan cara merinci dan menganalisis langkah-langkah selanjutnya dalam suatu prosedur atau sistem.

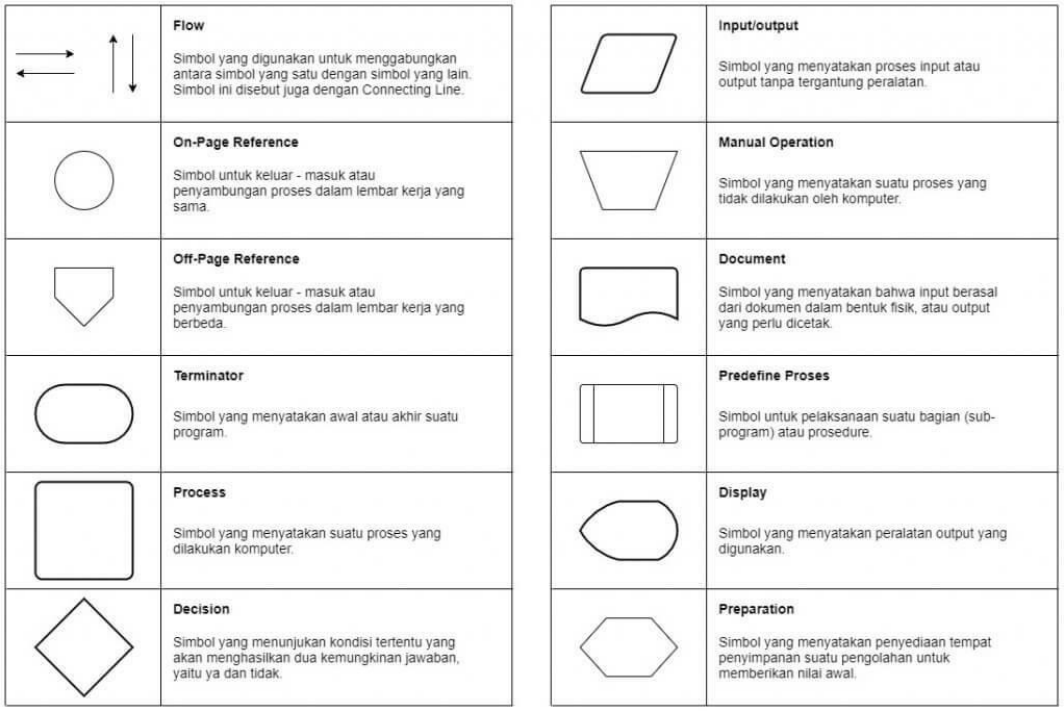
4. *Flowchart* Sistem

   Flowchart sistem adalah *flowchart* yang menampilkan tahapan atau proses kerja yang sedang berlangsung di dalam sistem secara menyeluruh. Selain itu *flowchart* sistem juga menguraikan urutan dari setiap prosedur yang ada di dalam sistem.

5. *Flowchart* Skematik

   Terakhir ada *flowchart* skematik. *Flowchart* ini menampilkan alur prosedur suatu sistem, hampir sama dengan *flowchart* sistem. Namun, ada perbedaan dalam penggunaan simbol-simbol dalam menggambarkan alur. Selain simbol-simbol, *flowchart* skematik juga menggunakan gambar-gambar komputer serta peralatan lainnya untuk mempermudah dalam pembacaan *flowchart* untuk orang awam.

Simbol-simbol dalam *flowchart* memiliki arti yang berbeda-beda. Berikut adalah simbol-simbol yang sering digunakan dalam proses pembuatan *flowchart* :



Gambar : 2.1 Simbol *Flowchart*

2.9. Pengertian UML

UML (*Unified Modelling Language*) adalah suatu metode dalam pemodelan secara visual yang digunakan sebagai sarana perancangan sistem berorientasi objek. Awal mulanya, UML diciptakan oleh *Object Management Group* dengan versi awal 1.0 pada bulan Januari 1997.UML juga dapat didefinisikan sebagai suatu bahasa standar visualisasi, perancangan, dan pendokumentasian sistem, atau dikenal juga sebagai bahasa standar penulisan *blueprint* sebuah *software*.

UML diharapkan mampu mempermudah pengembangan piranti lunak (RPL) serta memenuhi semua kebutuhan pengguna dengan efektif, lengkap, dan tepat. Hal itu termasuk faktor-faktor *scalability, robustness, security*, dan sebagainya.

Adapun tujuan dan fungsi perlu adanya UML yaitu sebagai berikut:

1. Dapat memberikan bahasa pemodelan visual atau gambar kepada para pengguna dari berbagai macam pemrograman maupun proses umum rekayasa.
2. Menyatukan informasi-informasi terbaik yang ada dalam pemodelan.
3. Memberikan suatu gambaran model atau sebagai bahasa pemodelan visual yang ekspresif dalam pengembangan sistem.
4. Tidak hanya menggambarkan model sistem software saja, namun dapat memodelkan sistem berorientasi objek.
5. Mempermudah pengguna untuk membaca suatu sistem.
6. Berguna sebagai blueprint, jelas ini nantinya menjelaskan informasi yang lebih detail dalam perancangan berupa coding suatu program.

UML juga dapat digunakan sebagai alat transfer ilmu tentang sistem aplikasi yang akan dikembangkan dari developer satu ke developer lainnya. UML sangat penting bagi sebagian orang karena UML berfungsi sebagai bridge atau jembatan penerjemah antara pengembang sistem dengan pengguna. Di sinilah pengguna dapat memahami sistem yang nantinya akan dikembangkan.

2.9.1. *Use Case* Diagram

Use Case diagram merupakan diagram yang menggambarkan hubungan antara aktor dengan sistem. *Use Case* diagram bisa

mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use Case* diagram juga bisa digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan bisa juga mempresentasikan sebuah interaksi aktor dengan sistem. Komponen tersebut kemudian menjelaskan komunikasi antara aktor, dengan sistem yang ada. Dengan demikian, *Use Case* dapat dipresentasikan dengan urutan yang sederhana, dan akan mudah dipahami oleh para konsumen. Manfaat dari *Use Case* sendiri adalah untuk memudahkan komunikasi dengan menggunakan *domain expert* dan juga *end user*, memberikan kepastian pemahaman yang pas tentang requirement atau juga kebutuhan sebuah sistem.

Use Case diagram mempunyai 3 komponen, yaitu :

1. Sistem

   Menyatakan batasan sistem dalam relasi dengan aktor-aktor yang menggunakannya (di luar sistem) dan fitur-fitur yang harus disediakan (dalam sistem).

2. Aktor

   Aktor adalah segala hal diluar sistem yang akan menggunakan sistem tersebut untuk melakukan sesuatu. Bisa merupakan manusia, sistem, atau device yang memiliki peranan dalam keberhasilan operasi dari sistem.

3. *Use Case*

   Use Case sendiri adalah gambaran fungsional dari sebuah sistem. Dengan demikian, antara konsumen dan juga pengguna pada sistem tersebut, akan mengerti atau paham mengenai fungsi sistem yang tengah dibangun.

Use Case diagram juga mempunyai beberapa relasi, yaitu:

1. *Association*

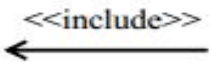
   Teknik mengindentifikasi interaksi yang dilakukan oleh *actor* tertentu dengan *Use Case* tertentu. Hal ini digambarkan dengan garis antara aktor terhadap *Use Case* tersebut.

2. *Generalization*

   Mendefinisikan relasi antara dua aktor atau dua *Use Case* yang mana salah satunya meng-*inherit* dan menambahkan atau *override* sifat dari yang lainnya.

3. *Dependency*

   Dependency ini terbagi menjadi 2 macam, yaitu *include* dan juga *extend*. Include berfungsi untuk mengindentifikasi hubungan antara 2 *Use Case*, dimana *Use Case* yang satu akan memanggil *Use Case* yang lainnya. *Extend* apabila pemanggilan, memerlukan kondisi tertentu maka akan berlaku dependensi.

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	*Association* : Abstraksi dari penghubung antara aktor dengan use case
	*Generalisasi* : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar: 2.2. Simbol Use Case Diagram (Redaksi Jagoan Hosting, 2022)

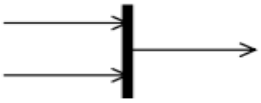
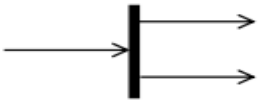
2.9.2. Activity Diagram

Activity Diagram merupakan rancangan aliran aktivitas atau aliran kerja dalam sebuah sistem yang akan dijalankan. *Activity Diagram* juga digunakan untuk mendefinisikan atau mengelompokan aluran tampilan dari sistem tersebut. *Activity Diagram* memiliki komponen

dengan bentuk tertentu yang dihubungkan dengan tanda panah. Panah tersebut mengarah ke-urutan aktivitas yang terjadi dari awal hingga akhir.

Fungsi dari *Activity Diagram* ini adalah sebagai berikut :

- Memperlihatkan urutan aktifitas proses pada sistem.
- Membantu memahami proses secara keseluruhan.
- *Activity Diagram* dibuat berdasarkan sebuah atau berapa *Use Case*.
- Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses.

NO	BENTUK SIMBOL	NAMA SIMBOL	FUNGSI SIMBOL
1.		Activity	Menyatakan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
2.		Control Flow	Menunjukan Urutan Eksekusi.
3.		Object Flow	Menunjukan aliran objek dari sebuah action atau activity ke action.
4.		Start Point	Menyatakan bahwa sebuah objek dibentuk atau diawali.
5.		End Point	Menyatakan bahwa sebuah objek dibentuk atau diakhiri.
6.		Join/Penggabungan	Menyatakan untuk menggabungkan kembali activity atau action yang parallel.
7.		Fork	Menyatakan untuk memecah behavior menjadi activity atau action yang parallel.
8.		Decision	Menunjukan penggambaran suatu keputusan/tindakan yang harus di ambil pada kondisi tertentu.

Gambar: 2.3. Simbol *Activity* Diagram (Repository BSI, 2022)

2.9.3. Sequence Diagram

Diagram sequence merupakan salah satu yang menjelaskan bagaimana suatu operasi itu dilakukan; *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.

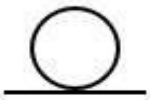
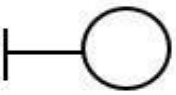
Diagram sequence menampilkan interaksi antar objek dalam dua dimensi. Dimensi vertikal adalah poros waktu, dimana waktu berjalan ke arah bawah. Sedangkan dimei horizontal merepresentasikan objek-objek individual. Tiap objek (termasuk *actor*) tersebut mempunyai waktu aktif yang direpresentasikan dengan kolom vertikal yang disebut dengan *lifeline*. Pesan (*message*) direpresentasikan sebagai panah dari satu *lifeline* ke *lifeline* yang lain. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, message akan dipetakan menjadi operasi/metoda dari *class*.

Diagram sequence mendeskripsikan bagaimana entitas dalam sistem berinteraksi, termasuk pesan yang digunakan saat interaksi. Semua pesan dideskripsikan dalam urutan dari eksekusi. Diagram *sequence* berhubungan erat dengan diagram *Use Case* , dimana satu *Use Case* akan menjadi satu *diagram sequence*.

Diagram sequence memiliki elemen-elemen sebagai berikut:

1. Actor, yaitu orang atau sistem eksternal lainnya yang menerima manfaat atau menggunakan sistem.

2. Object, yaitu objek yang terlibat dalam sistem.
3. Lifeline, yaitu sebuah garis yang menggambarkan masa hidup dari sebuah objek dalam *sequence* diagram.
4. *Execution occurrence*, yaitu sebuah persegi panjang yang menggambarkan waktu terjadinya pengiriman/penerimaan pesan.
5. *Message* yaitu informasi yang mengalir dari satu objek ke objek lainnya.
6. *Guard condition* yaitu suatu persyaratan yang harus terpenuhi agar suatu pesan dapat dikirimkan.
7. *Object destruction* menggambarkan akhir dari sebuah *lifeline object*.
8. *Frame* menyatakan konteks dari *diagram sequence*.

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menggambar orang yang sedang berinteraksi dengan sisitem.
2		*Entity Class*	Menggambarkan hubungan yang akan dilakukan
3		*Boundary Class*	Menggambarkan sebuah gambbaran dari foem
4		*Control Class*	Menggambarkan penghubung antara boundary dengan tabel
5		*A focus of Control & A Life Line*	Menggambarkan tempat mulai dan berakhirnya massage
6		*A massage*	Menggambarkan Pengiriman Pesan

Gambar: 2.4. Simbol *Sequence* Diagram ("Simbol *Sequence* Diagram - Badoy Studio," 2020)

2.9.4. Class Diagram

Class diagram adalah salah satu jenis diagram berbentuk struktur pada model UML. Diagram ini menggambarkan struktur, atribut, kelas, hubungan dan metode dengan sangat jelas dari setiap objeknya.

Diagram kelas memberikan data berupa hubungan apa yang terjadi diantara kelas-kelas, bukan menjelaskan kejadiannya. *Class diagram* dalam suatu proyek umumnya menggunakan konsep yang disebut *object- oriented*, sehingga membuatnya mudah untuk digunakan. Berikut adalah gambar dari simbol *class diagram*

NO	BENTUK SIMBOL	NAMA SIMBOL	FUNGSI SIMBOL
1.		Kelas	Kelas pada struktur sistem
2.		Antarmuka/*interface*	Sama dengan konsep interface dalam pemrograman berorientasi objek
3.		Asosiasi/*association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
4.		Asosiasi berarah/directed association	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
5.		Generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
6.		Kebergantungan/depedency	Relasi antar kelas dengan kebergantungan antar kelas
7.		Agregasi/aggregation	Relasi antar kelas dengan makna semua bagian (whole-part)

Gambar: 2.5. Simbol *Class* Diagram (Repositori BSI, 2022)

2.10. Pengertian *Entity Relationship* Diagram

Entity Relationship Diagram adalah salah satu jenis diagram yang sifatnya lebih struktural dan bisa digunakan untuk dimanfaatkan dalam suatu desain pada suatu database ataupun pada sebuah business plan. Terdapat banyak sekali komponen yang terdapat di dalam susunan *Entity Relationship diagram*, beberapa diantaranya adalah sebagai berikut:

1. Entitas

   Entitas adalah sekumpulan objek yang nantinya akan diidentifikasi. Ketika sedang membuat ERD, umumnya suatu entitas akan digambarkan dalam suatu simbol persegi panjang. Disisi lain, entitas yang lemah akan digambarkan dengan simbol persegi panjang yang kecil di dalam persegi panjang yang lebar. Masing-masing entitas sudah tentu mempunyai perbedaan. Bila ternyata ada kesamaan, maka entitas tersebut tidak perlu dicantumkan.

2. Atribut

   Setiap entitas selalu mempunyai elemen ataupun atribut agar bisa menjelaskannya dengan baik. Umumnya, atribut akan digambarkan sebagai simbol. Di dalam *Entity Relationship diagram*, terdapat beberapa jenis atribut, seperti atribut simple, atribut kunci, atribut multinilai, atribut gabungan, dan juga atribut derivatif.

3. Relasi

   Relasi ataupun hubungan adalah suatu tingkat ketertarikan pada beberapa entitas dari gabungan lainnya. umumnya, relasi ini akan dicerminkan dalam simbol berbentuk belah ketupat. Di dalam *Entity Relationship* diagram, relasi ini akan dibagi menjadi beberapa jenis, seperti One to One, One to Many, dan juga Many to Many.

Entity Relationship Diagram (ERD) didesain dengan berbagai fungsi dan tujuannya masing-masing. Namun, fungsi umum dari *Entity Relationship* diagram adalah sebagai berikut:

- Membantu menganalisis suatu database dengan cara yang lebih cepat dan juga lebih murah.
- Mampu menjalankan relasi antar setiap data yang mempunyai keterkaitan dengan berdasarkan objek yang dihubungkan dengan suatu relasi khusus.
- Membantu menjalankan dokumentasi data yang terdapat dalam suatu database dengan cara melakukan analisis dan identifikasi pada setiap objek ataupun entitas serta relasinya.
- Melakukan suatu pengujian model yang sebelumnya sudah dibuat.

NO	BENTUK SIMBOL	NAMA SIMBOL	FUNGSI SIMBOL
1		Entitas	Menunjukan suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
2		Relasi	Menunjukan adanya hubungan di antara sejumlah entitas yang berbeda
3		Atribut	Berfungsi mendeskripsikan karakter entitas (atribut yang berfungsi sebagai key diberi garis bawah)
4		Garis	Menunjukan penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

Gambar: 2.6. Simbol ER Diagram (Repository BSI, 2022)

BAB III

ANALISA MASALAH DAN PERANCANGAN PROGRAM

Dalam penelitian ini digunakan metode perancangan dengan model *Prototype*. Umumnya model *prototype* terdiri dari tujuh tahapan yaitu pengumpulan kebutuhan, membangun *prototyping*, evaluasi *prototype*, mengkodekan sistem, menguji sistem, evaluasi sistem, dan menggunakan sistem. Namun, pada penelitian ini hanya akan menggunakan enam tahapan dari model *prototype*, yaitu pengumpulan kebutuhan, membangun *prototyping*, evaluasi *prototype*, mengkodekan sistem, menguji sistem, dan evaluasi sistem.

3.1. Studi Literatur

Studi Literatur merupakan teknik yang digunakan untuk memperoleh informasi atau data dengan mempelajari buku-buku serta jurnal yang berkaitan dengan perancangan aplikasi berbasis android menggunakan metode *Reverse Geocoding*. Berikut adalah tabel 3.1. yang menjadi referensi penelitian :

Tabel 3.1. Studi Literatur

No	Penulis	Judul	Pembahasan
1.	Iqbal Dwiki Kurniawan, Dwi Sunaryono, dan Adhatus Solichah Ahmadiyah (2014)	Aplikasi Monitoring Keberadaan Objek Melalui Perangkat Bergerak Berbasis *Android*	Aplikasi yang dibuat untuk memonitoring objek dengan memanfaatkan penggunaan GPS, *Google Maps* serta layanan pesan singkat yang diaplikasikan pada perangkat bergerak. Objek atau pengguna dapat mengetahui lokasi terkini pada aplikasi tersebut atau bisa mendapatkan *link* posisi pengguna yang dikirimkan melalui layanan pesan singkat. Dengan digunakannya metode *Reverese Geocoding* dan *broadcast-receiver*, hasil pengujian menujukkan bahwa 100% pengguna setuju implementasi dari metode *Reverse Geocoding* dapat menujukkan posisi aktual dari pengguna aplikasi.

No	Penulis	Judul	Pembahasan
2.	Risa Herdianto, Eko Budi Setiawan	*Android Mobile Application Development For Field Visit Documentation.*	Aplikasi yang dibuat untuk membantu dalam pendokumentasian karyawann Dinas Pangan Kabupaten Majalengka dalam melakukan kunjungan dan peninjauan langsung ke lapangan. Dokumentasi yang diperlukan saat kunjangan dan peninjauan langsung ke lapangan adalah bukti foto, lokasi dan suhu. Aplikasi ini akan mengambil gambar dimana di dalamnya memuat informasi seperti suhu dan lokasi. Hasil pengujian aplikasi ini menunjukkan bahwa aplikasi berjalan dengan baik dengan menghasilkan dokumentasi berupa foto yang memuat informasi berupa suhu dan lokasi saat pengambilan foto tersebut.

No	Penulis	Judul	Pembahasan
3.	Yusuf Safrudin (2018)	Penerapan *Location Based Services* Untuk Aplikasi *Tracking* Lari Berbasis *Mobile*	Aplkasi yang dibuat untuk memonitoring seberapa jauh olahraga lari yang dilakukan oleh pengguna sehingga dapat mengukur seberapa jauh lari yang sudah dilakukan, dan pengguna dapat mengetahui batasan kemampuan fisik dari pengguna dilihat dari jarak tempuh lari yang dilakukan. Dengan menerapkan *Location Based Services*, aplikasi akan mulai melakukan pencatatan posisi awal lari sampai selesai, yang selanjutnya akan diukur jarak antara titik awal dan titik akhir dan dihitungkan kalori yang sudah dikeluarkan pengguna.
4.	Suratno (2019)	Penerapan *Reverse Geocoding* Untuk Aplikasi Absensi	Aplikasi yang dibuat sebagai absensi *mobile* karyawan PT. Kencana Alam Putra yang

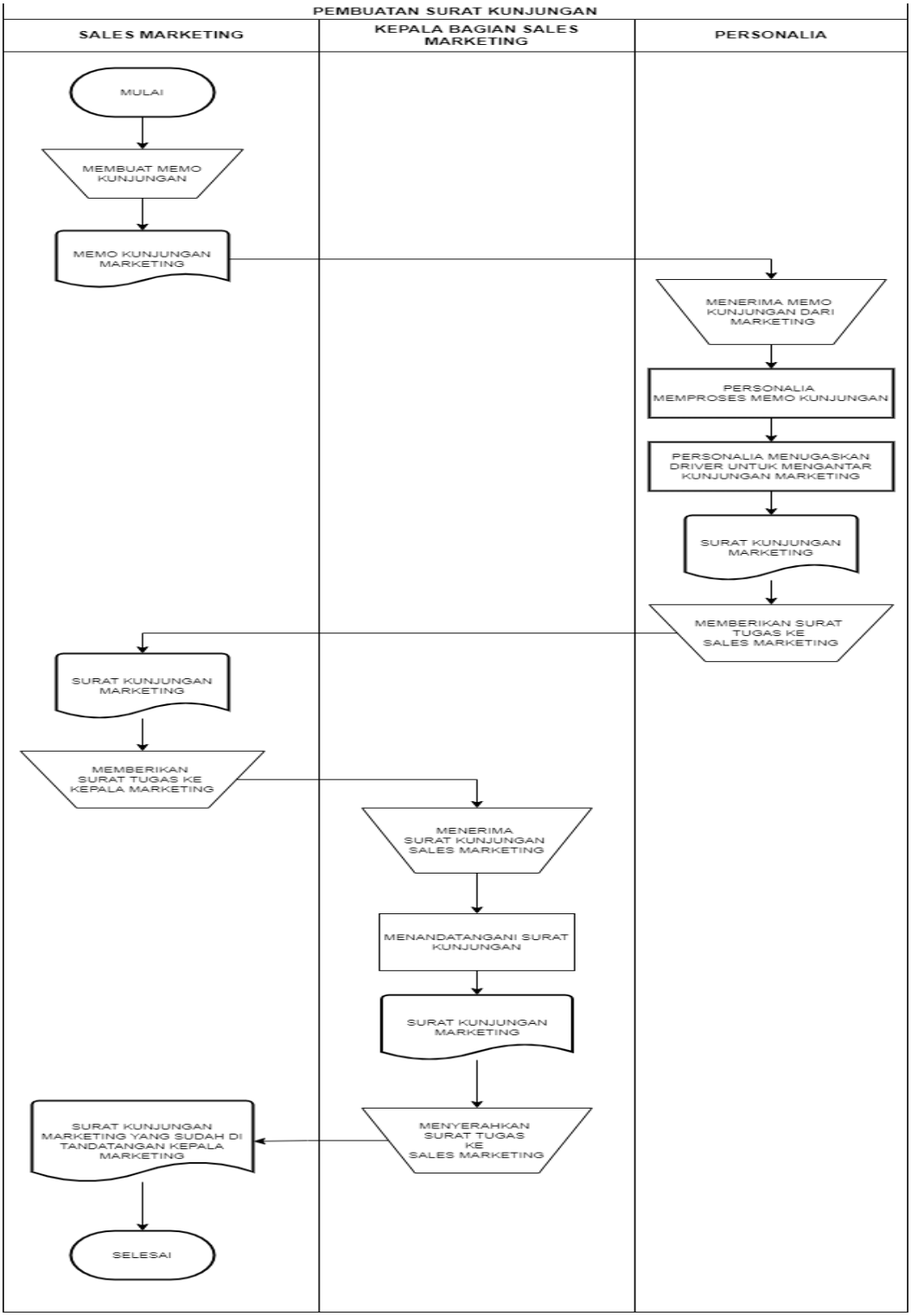
No	Penulis	Judul	Pembahasan
		Mobile Di PT. Kencana Alam Putra	tidak sedang berada di dalam kantor dengan mengirimkan foto dan lokasi karyawan. Dengan menerapkan *Reverse Geocoding*, aplikasi akan mengambil titik koordinat lokasi dan mengubahnya menjadi alamat yang dimengerti oleh pengguna. Hasil penelitian menujukkan bahwa aplikasi berjalan sesuai yang diharapkan dan mempermudah karyawan dalam hal absensi, serta mempermudah bagian HRD dalam melakukan pemantauan dan penilaian kinerja karyawan.
5.	Bambang Riono, Widia Rifkianti	*Android-Based Information System Of Online Teaching Services With Geo-*	Sistem Informasi Layanan Pengajaran Online dengan *geo-location* adalah sistem yang menyatukan dan mengatur

No	Penulis	Judul	Pembahasan
		Location Determination.	pertemuan antara guru privat dan orang yang mencari guru privat melalui aplikasi Android yang didukung dengan geo-lokasi. Orang-orang dapat mencari tutor mata pelajaran terkait untuk waktu pertemuan dan lokasi tertentu, lalu sistem akan mencari tutor dengan waktu ketersediaan yang cocok dalam radius lokasi terdekat. Selain itu, siswa dapat mengatur atau merencanakan sesi pertemuan pembelajaran dengan guru melalui aplikasi. Sistem informasi ini berbasis *client-server* dimana dibangun aplikasi terdiri dari aplikasi Android yang berjalan di lingkungan klien dan aplikasi *web* yang berjalan di lingkungan *server*. Aplikasi

No	Penulis	Judul	Pembahasan
			web berfungsi sebagai layanan *web* dan antarmuka bagi administrator untuk mengelola *data master*, melihat pengguna dan data transaksi, mencetak laporan, dan menanggapi keluhan dari pengguna akhir.

3.2. Pengumpulan Kebutuhan

Pada tahapan ini, Divisi HRD dan Divisi IT bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat. Hal ini dilakukan mengingat sistem yang sebelumnya yang mana dalam hal pengurusan *visit* dilakukan secara *offline* atau menggunakan media kertas berupa surat. Berikut adalah diagram *flowchart* sistem yang berjalan sebelumnya :



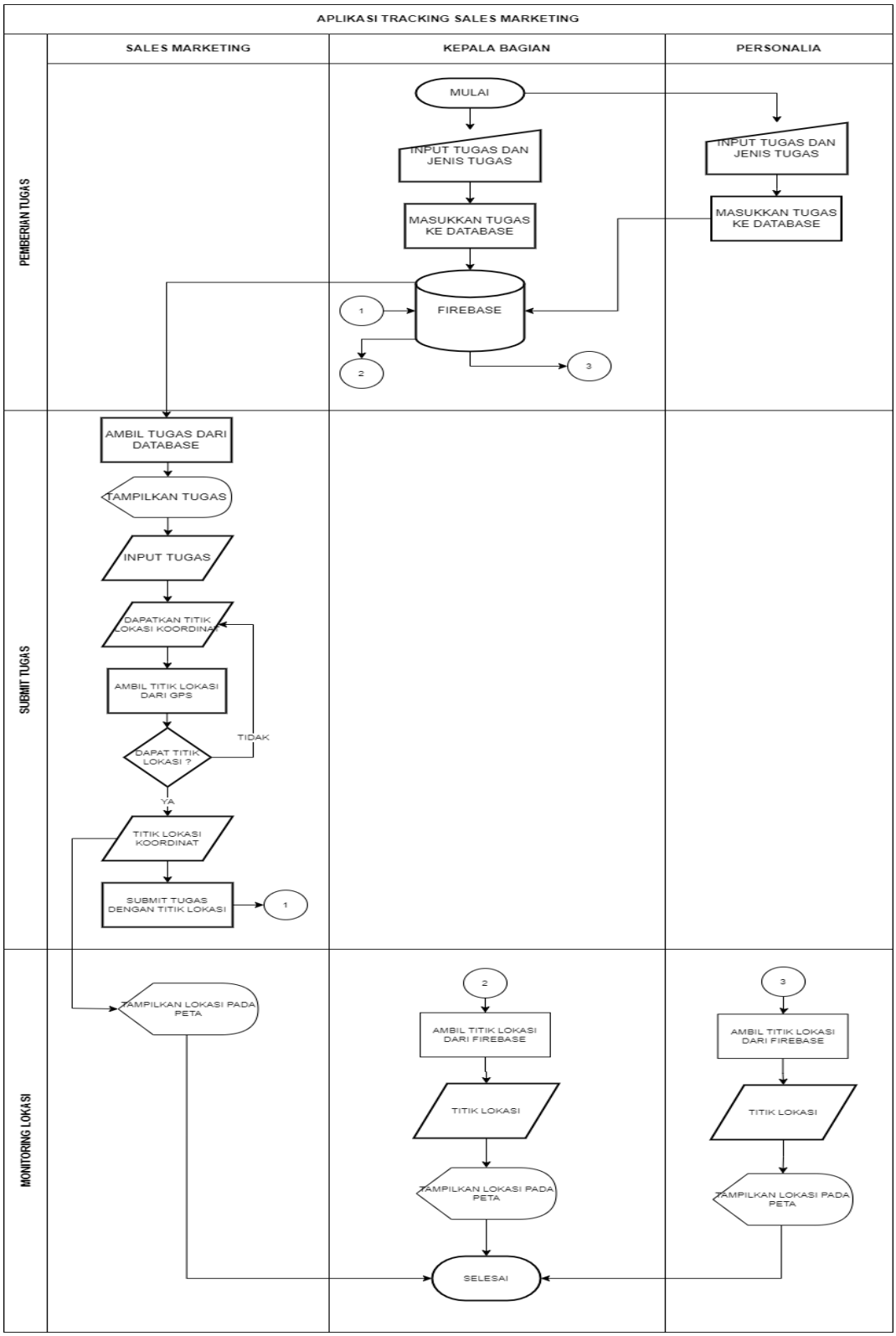
Gambar: 3.1. *Flowchart* Pembuatan Memo Kunjungan Kepala Bagian Marketing

3.2.1. Analisa Masalah

Permasalahan yang sedang terjadi di PT. Sri Indah Labetama, dimana kepala bagian divisi marketing dan bagian personalia kesulitan untuk memantau dan memonitoring secara langsung *Sales Marketing* yang sedang melakukan *visit* ke *customer*, sehingga diusulkan sebuah rancangan aplikasi *tracking* agar kepala bagian divisi *Sales Marketing* dapat memantau dan memonitoring secara langsung posisi dari *Sales Marketing*nya.

3.2.2. Pengusulan Sistem

Dari analisa masalah diatas, perlu dibuat sebuah aplikasi yang digunakan baik oleh kepala bagian divisi marketing, bagian personalia, dan *Sales Marketing* sendiri. Aplikasi ini nantinya akan mengirimkan tugas apa saja yang akan dilakukan oleh *Sales Marketing*. Dari aplikasi *Sales Marketing* akan meminta titik koordinat lokasi saat ini dan akan diubah menjadi alamat lokasi yang lebih mudah baca yang nantinya akan di unggah ke basis data. Dari aplikasi kepala bagian divisi marketing dan bagian personalia akan mengambil data dari basis data berupa alamat lokasi yang sudah diunggah oleh *Sales Marketing*. Proses ini akan digambarkan dalam diagram *flowchart* sebagai berikut :



Gambar: 3.2. *Flowchart* Sistem Yang Diusulkan

3.2.3. Analisa Kebutuhan Sistem

Sistem yang akan dibangun adalah aplikasi android menggunakan Bahasa pemrograman Java dan XML sehingga diperlukan beberapa kebutuhan untuk melakukan pembangunan aplikasi android. Berikut adalah kebutuhan minimum yang dibutuhkan dalam pembuatan aplikasi *tracking* ini :

- *Hardware* (Perangkat Keras)

    1. Komputer / Laptop dengan spesifikasi minimum :

        - CPU dengan arsitektur 64 bit; Intel Core generasi ke dua atau yang terbaru, atau AMD CPU yang mendukung *Windows Hypervisor* untuk OS *Windows, Hypervisor Framework* untuk OS MacOS, dan *AMD Virtualization (AMD-V)* dan *SSSE3* untuk OS Linux
        - Kapasitas RAM 8 GB atau lebih
        - Harddisk dengan minimum penyimpanan yang tersedia 8 GB (IDE + Android SDK + Android Emulator

    2. *Keyboard*
    3. *Mouse*
    4. *Monitor, dengan resolusi minimum 1280 X 800*

- *Software* (Perangkat Lunak)

    1. *Operating System*, dengan jenis minimum sebagai berikut :

        - *64-bit Microsoft® Windows® 8/10/11*
        - *MacOS® 10.14 (Mojave)* atau diatasnya
        - Semua distro *64-bit Linux* yang support *Gnome, KDE, or Unity DE; GNU C Library (glibc) 2.31 or later*

    2. *Android Studio*
    3. *Firebase*

3.3. Membangun Prototype

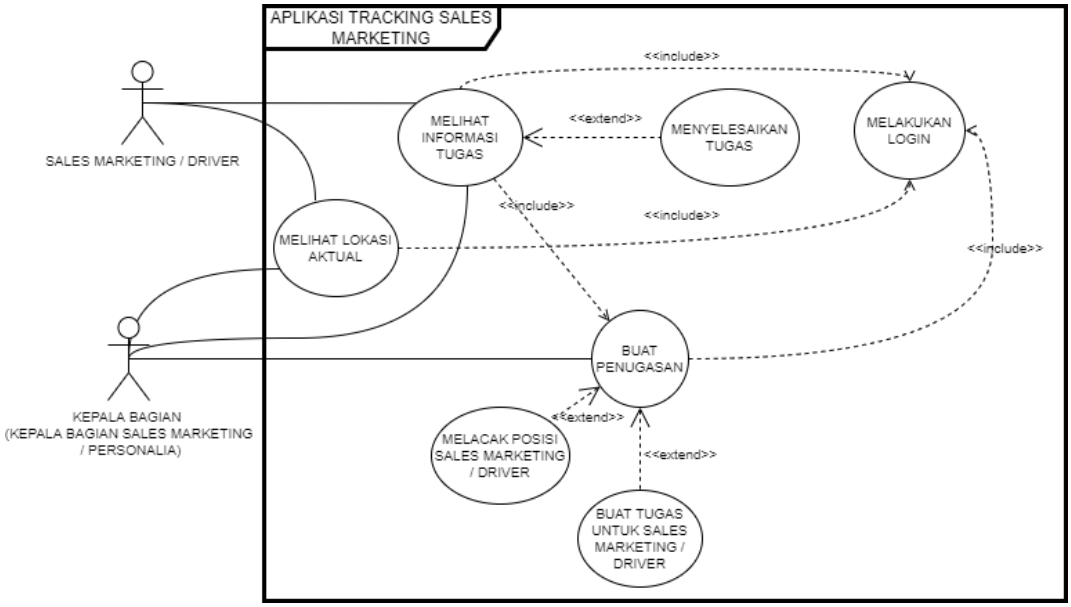
Setelah melakukan pengumpulan kebutuhan, tahapan penelitian selanjutnya adalah membangun *prototype*, dimana pada tahapan ini akan dilakukan perancangan yang sifatnya sementara dengan fokus utama penyajian kepada perusahaan.

3.3.1. *Use Case* Diagram

Use Case diagram akan menggambarkan hubungan antara pengguna dengan sistem, dimana pada diagram akan ada *actor* sebagai pengguna dan *Use Case* sebagai aktivitas yang terdapat di dalam aplikasi.

Tabel 3.2. Tabel Deskripsi Aktor

No	Aktor	Deskripsi
1.	*Sales Marketing* / *Driver*	Pengguna aplikasi yang hanya bisa menjalankan tugas yang sudah ditugaskan oleh kepala divisi *Sales Marketing* serta hanya bisa melihat posisi lokasi pengguna saja
2.	Kepala Bagian (Kepala *Sales Marketing* / Bagian Personalia)	Pengguna aplikasi yang bisa memberikan tugas kepada *Sales Marketing* serta dapat memantau lokasi terkini *Sales Marketing*



Gambar: 3.5. *Use Case* Aplikasi *Tracking* Dengan *Actor Sales Marketing / Driver*

Tabel 3.3. Tabel Skenario *Use Case Login*

Nama *Use Case*	Login
Aktor	*Sales Marketing* / Driver
Skenario	1. Input alamat email dan password 2. Menekan tombol login

Tabel 3.4. Tabel Skenario *Use Case* Informasi Tugas

Nama *Use Case*	Informasi Tugas
Aktor	*Sales Marketing* / Driver
Skenario	1. Ditampilkan tugas yang diberikan oleh kepala bagian *Sales Marketing* atau bagian personalia. 2. *Sales Marketing* / Driver men-klik kolom Jam Masuk sebagai awal mula pembaharuan lokasi dan tugas yang dikerjakan dimulai 3. *Sales Marketing* / Driver men-klik kolom Jam Keluar sebagai berakhirnya pembaharuan lokasi. 4. Jika tidak ada tugas akan ditampilkan tampilan "Selamat! Tugas anda selesai"

Tabel 3.5. Tabel Skenario *Use Case Submit* Tugas

Nama *Use Case*	Submit Tugas
Aktor	*Sales Marketing* / Driver
Skenario	1. *Sales Marketing* menerima list tugas yang sudah diinput oleh kepala bagian *Sales Marketing* atau bagian personalia 2. Jika sudah menyelesaikan salah satu tugas, *Sales Marketing* / Driver akan men-swipe tugas tersebut 3. Tugas yang sudah di swipe akan diperbaharui statusnya di server sebagai selesai bersamaan dengan lokasi terakhir saat tugas di swipe

Tabel 3.6. Tabel Skenario *Use Case* Posisi Terkini

Nama *Use Case*	Posisi Terkini
Aktor	*Sales Marketing* / Driver
Skenario	1. *Sales Marketing* menekan tab “Lokasi saya” *2.* Map akan memunculkan posisi terkini *Sales Marketing* / Driver

Tabel 3.7. Tabel Skenario *Use Case Login*

Nama *Use Case*	Login
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. Input alamat email dan password 2. Menekan tombol login

Tabel 3.8. Tabel Skenario *Use Case* Informasi Tugas

Nama *Use Case*	Informasi Tugas
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. Ditampilkan tugas yang diberikan oleh kepala bagian *Sales Marketing* atau bagian personalia. 2. Kepala bagian *Sales Marketing* men-klik kolom Jam Masuk sebagai awal mula pembaharuan lokasi dan tugas yang dikerjakan dimulai 3. Kepala bagian *Sales Marketing* men-klik kolom Jam Keluar sebagai berakhirnya pembaharuan lokasi. 4. Jika tidak ada tugas akan ditampilkan tampilan "Selamat! Tugas anda selesai"

Tabel 3.9. Tabel Skenario *Use Case Submit* Tugas

Nama *Use Case*	Submit Tugas
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. Kepala bagian *Sales Marketing* menerima list tugas yang sudah diinput. 2. Jika sudah menyelesaikan salah satu tugas, kepala bagian *Sales Marketing* akan men-swipe tugas tersebut 3. Tugas yang sudah di swipe akan diperbaharui statusnya di server sebagai selesai bersamaan dengan lokasi terakhir saat tugas di swipe

Tabel 3.10. Tabel Skenario *Use Case* Posisi Terkini

Nama *Use Case*	Posisi Terkini
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. *Sales Marketing* menekan tab “Lokasi saya” *2.* Map akan memunculkan posisi terkini *Sales Marketing* / Driver

Tabel 3.11. Tabel Skenario *Use Case* Buat Penugasan

Nama *Use Case*	Buat Penugasan
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. Kepala Bagian *Sales Marketing* menekan floating button

Tabel 3.12. Tabel Skenario *Use Case* Tambah Tugas

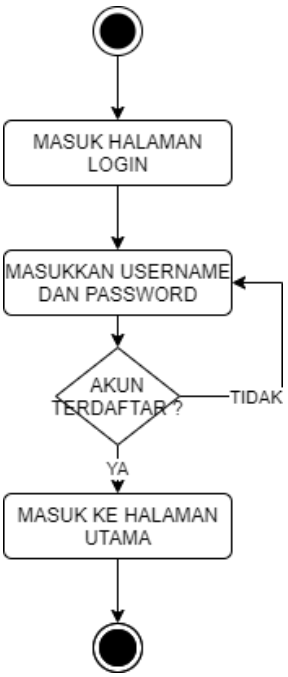
Nama *Use Case*	Tambah Tugas
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. Menginput Nama Karyawan 2. Memilih Jenis Tugas 3. Menginput alamat 4. Klik tombol Input Tugas

Tabel 3.13. Tabel Skenario *Use Case* Posisi Karyawan

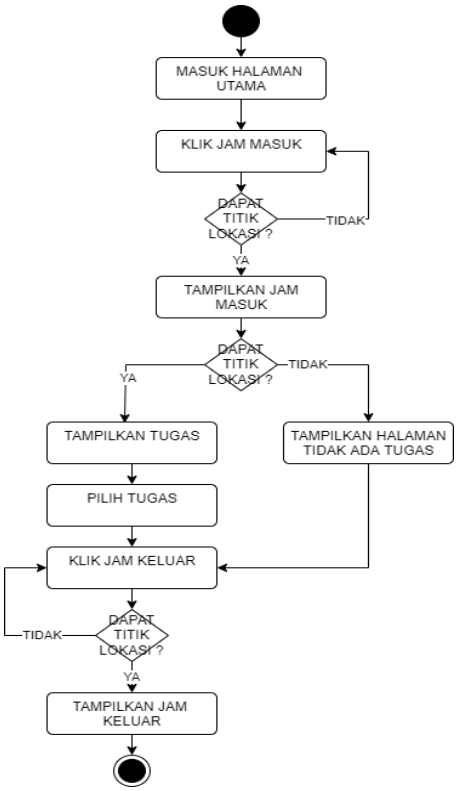
Nama *Use Case*	Posisi Karyawan
Aktor	Kepala Bagian *Sales Marketing*
Skenario	1. Input pencarian nama *Sales Marketing* / Driver 2. Ditampilkan marker pada peta lokasi yang sudah dikunjungi oleh *Sales Marketing* / Driver

3.3.2. *Activity* Diagram

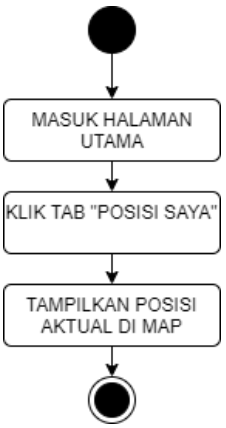
Activity Diagram merupakan rancangan aliran aktivitas atau aliran kerja dalam sebuah sistem yang akan dijalankan. *Activity* Diagram juga digunakan untuk mendefinisikan atau mengelompokan aluran tampilan dari sistem tersebut. *Activity* Diagram memiliki komponen dengan bentuk tertentu yang dihubungkan dengan tanda panah. Panah tersebut mengarah ke-urutan aktivitas yang terjadi dari awal hingga akhir.



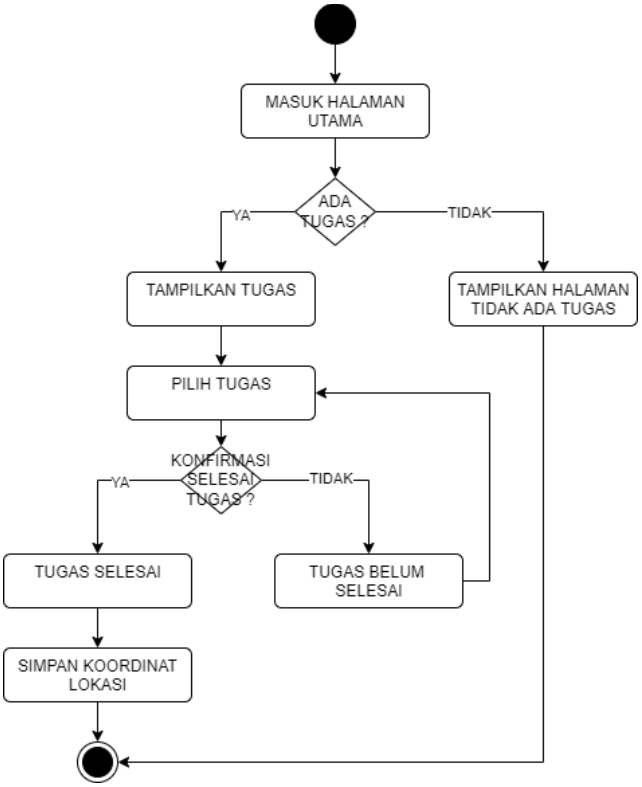
Gambar: 3.6. *Activity* Diagram Login *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



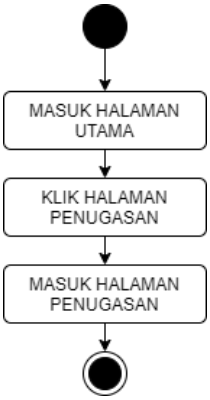
Gambar: 3.7. *Activity Diagram* Tampilkan Jam Masuk dan Keluar *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



Gambar: 3.8. *Activity Diagram* Lokasi saya *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



Gambar: 3.9. *Activity Diagram* Pilih Tugas *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



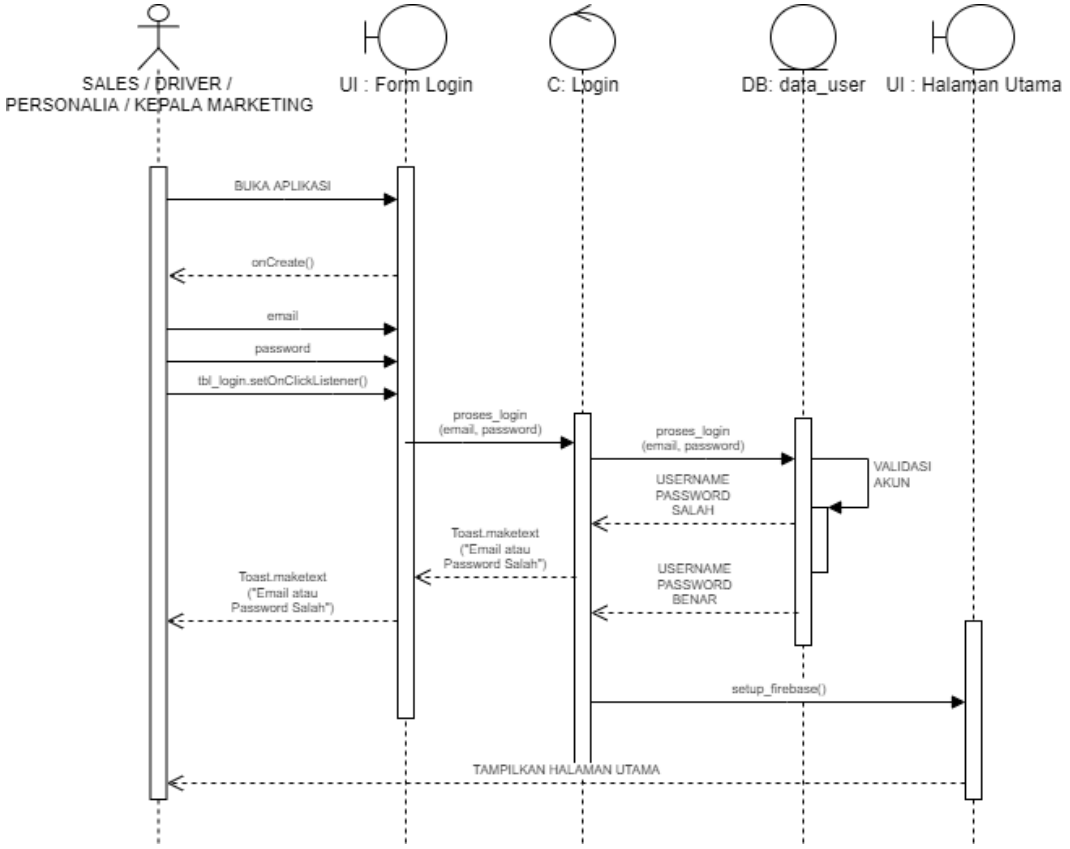
Gambar: 3.10. *Activity Diagram* Akses Halaman Penugasan Kepala Bagian *Sales Marketing* / Personalia

Gambar: 3.11. *Activity Diagram* Pembuatan Tugas Kepala Bagian *Sales Marketing* / Personalia

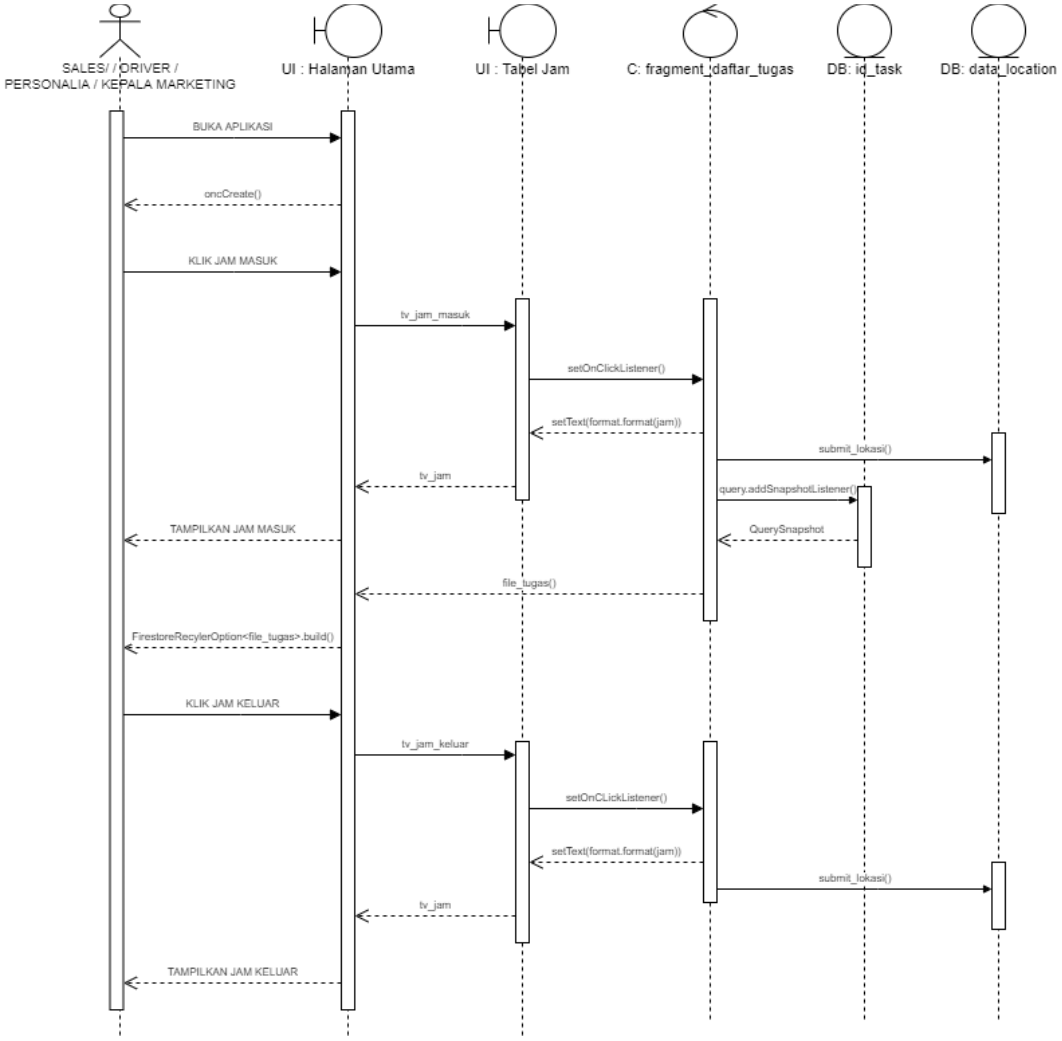
Gambar 3.12. *Activity* Diagram Pantau Akses Lokasi Kepala Bagian *Sales Marketing* / Personalia (Kanan)

3.3.3. *Sequence* Diagram

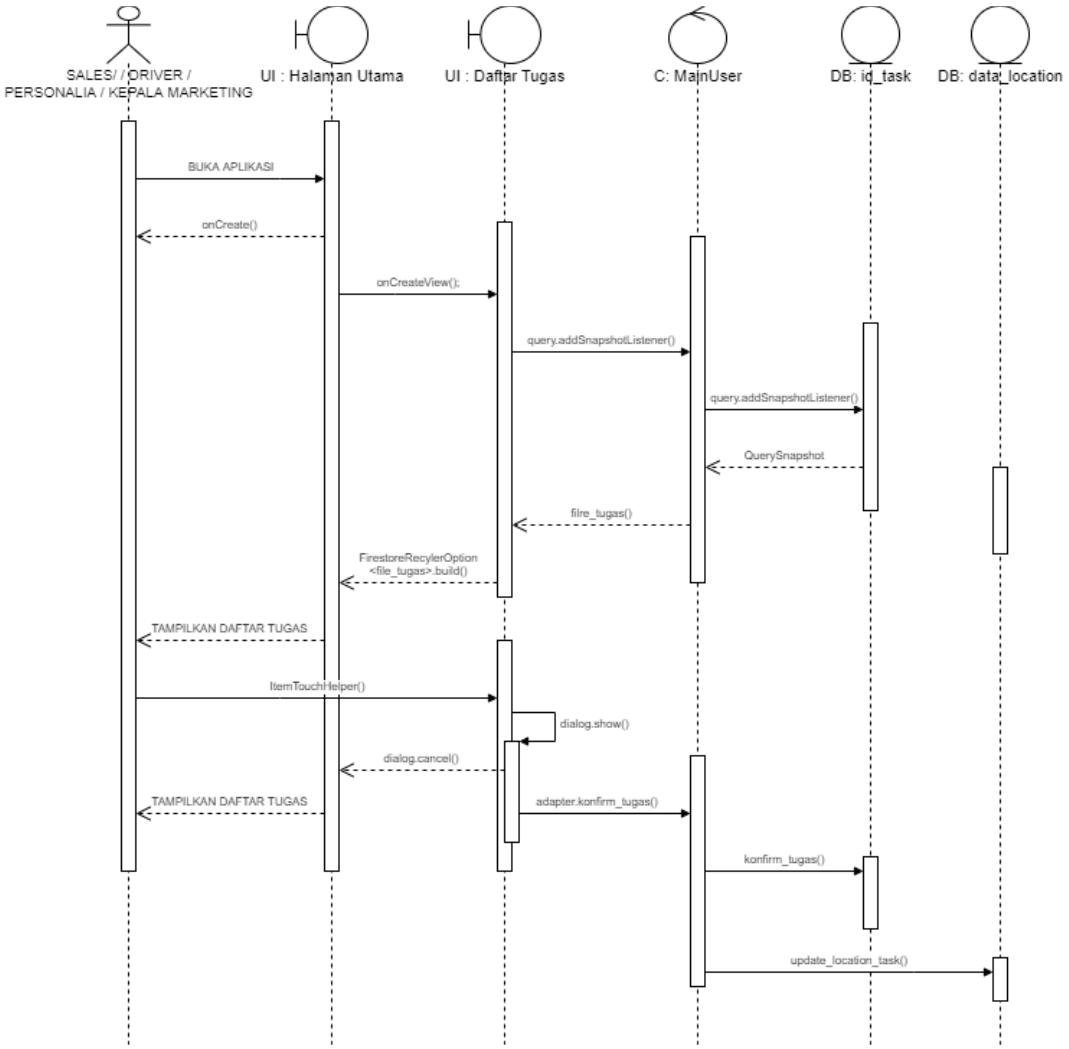
Diagram *sequence* merupakan salah satu yang menjelaskan bagaimana suatu operasi itu dilakukan; *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.



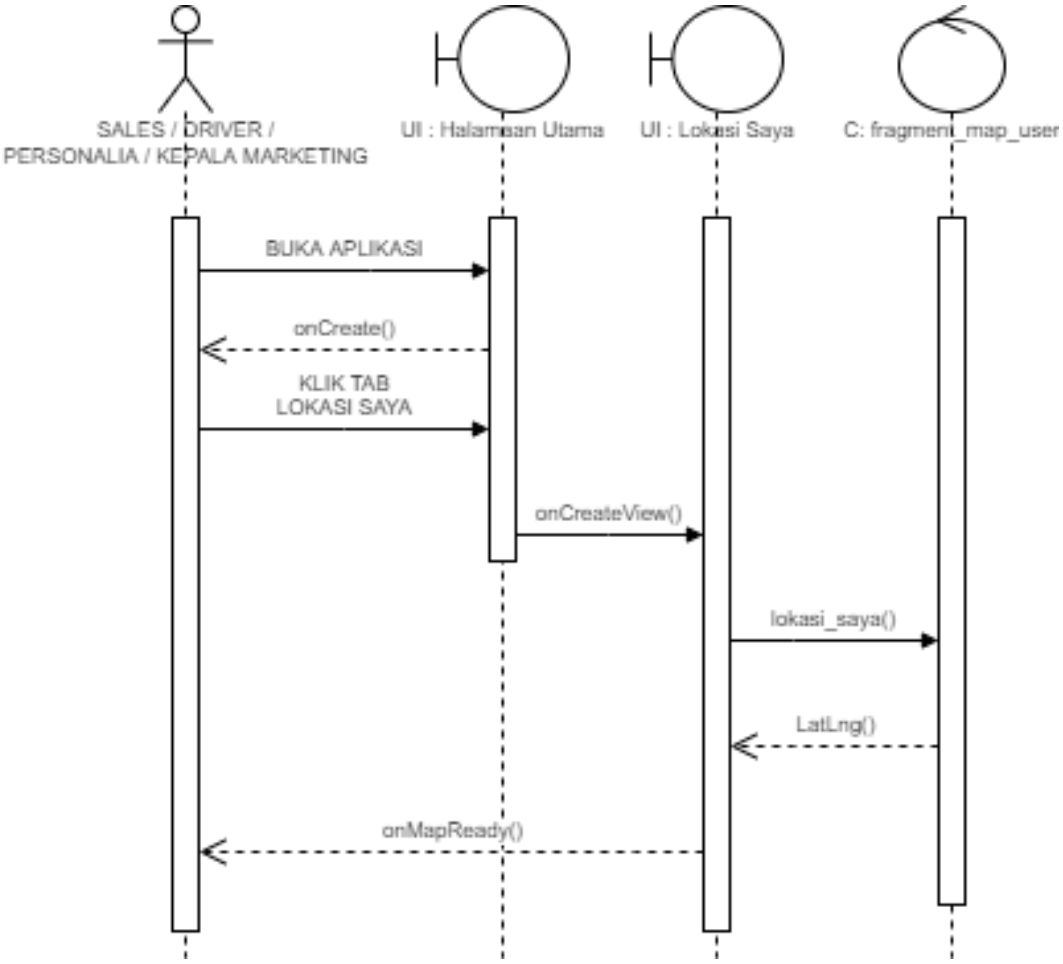
Gambar: 3.13. *Sequence* Diagram Login *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



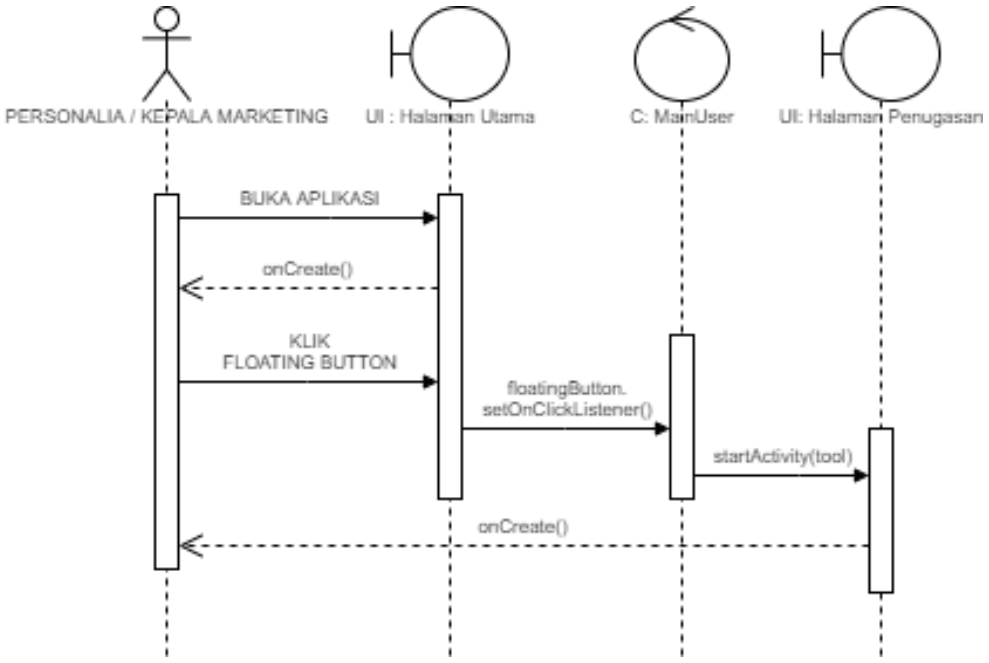
Gambar: 3.14 *Activity* Diagram Informasi Tugas *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



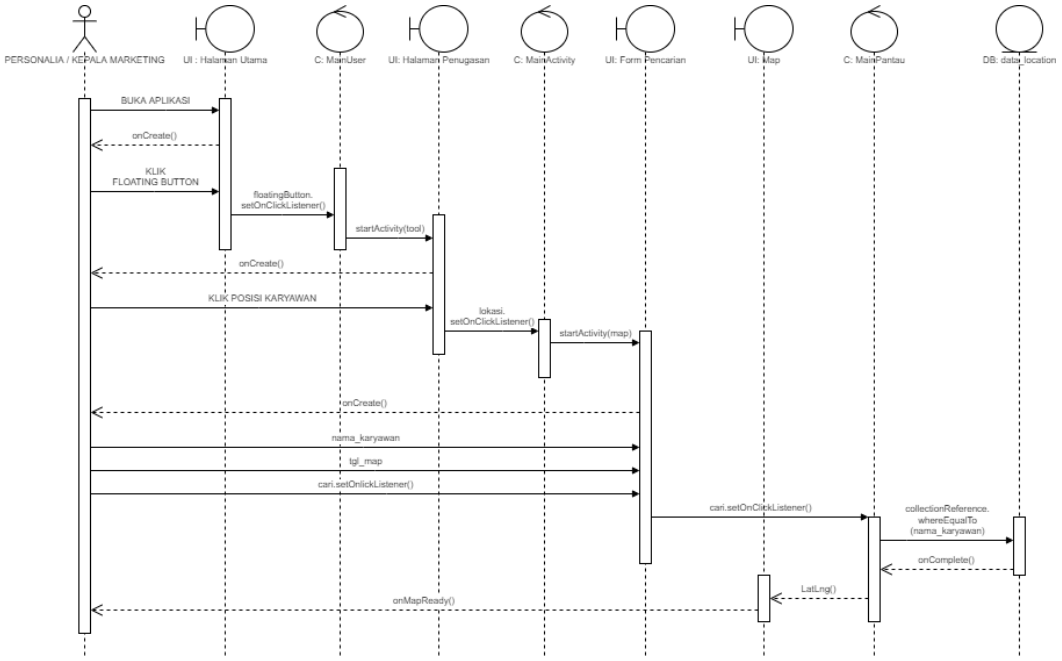
Gambar: 3.15. *Sequence* Diagram Submit Tugas *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



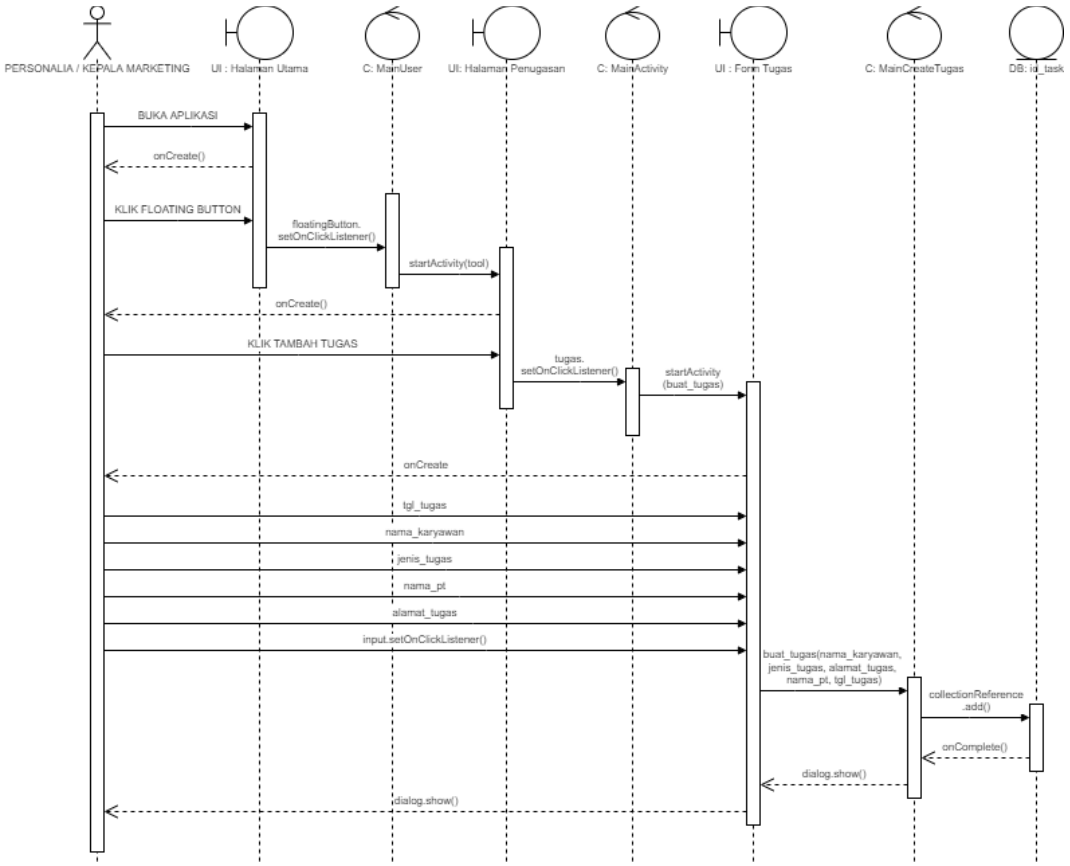
Gambar: 3.16. *Sequence* Diagram Lokasi Saya *Sales Marketing* / Driver / Kepala Bagian *Sales Marketing* / Personalia



Gambar: 3.17. *Sequence* Diagram Halaman Penugasan Kepala Bagian *Sales Marketing* / Personalia



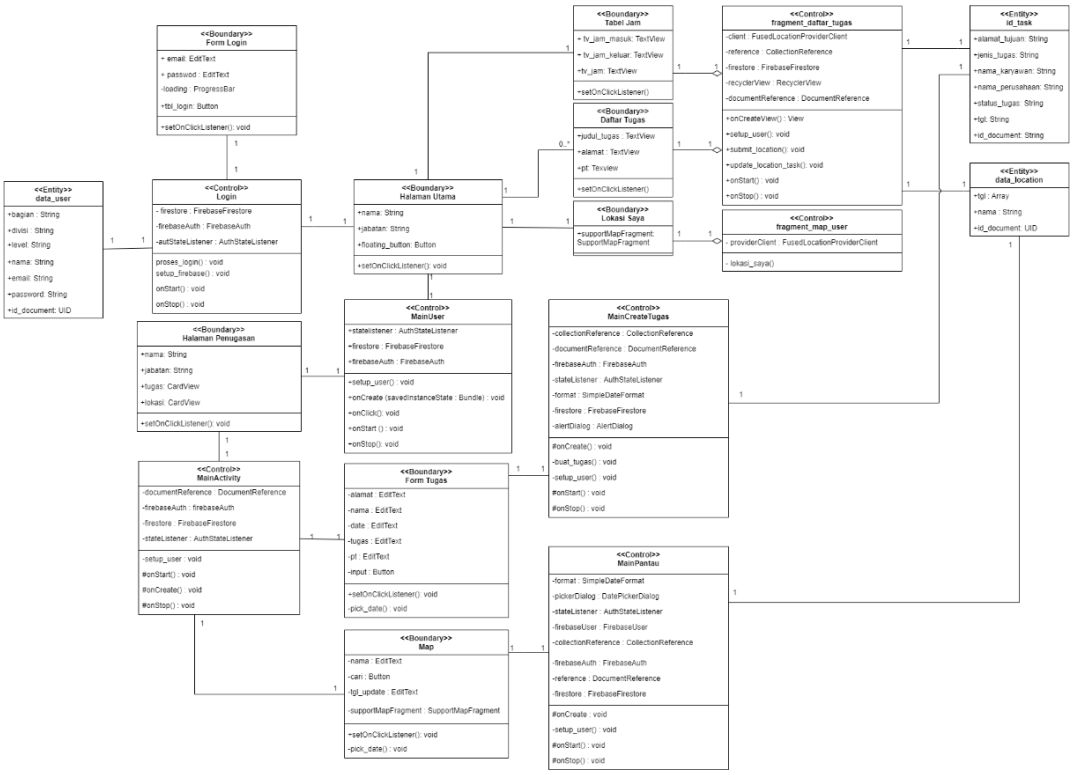
Gambar: 3.18. *Sequence* Diagram Pantau Lokasi Kepala Bagian *Sales Marketing* / Personalia



Gambar: 3.19. *Sequence* Diagram Buat Tugas Kepala Bagian *Sales Marketing* / Personalia

3.3.4. *Class* Diagram

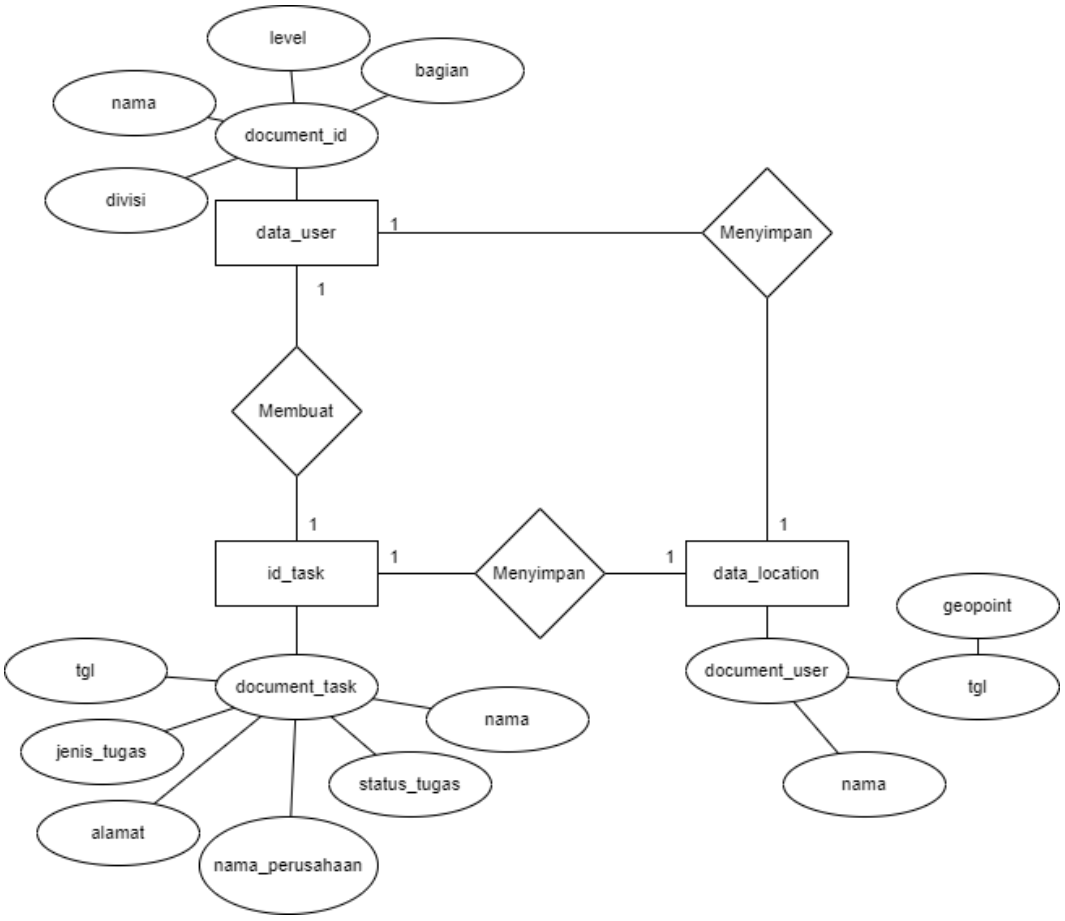
Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi *class*, atribut, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi.



Gambar: 3.20. *Class* Diagram Aplikasi *Tracking Sales Marketing*

3.3.5. *Entity Relationship* Diagram

ERD (*Entity Relationship Diagram*) atau diagram hubungan entitas adalah diagram yang digunakan untuk perancangan suatu *database* dan menunjukan relasi antar objek atau entitas beserta atribut-atributnya secara detail.



Gambar: 3.21. ER Diagram Aplikasi *Tracking Sales Marketing*

3.3.6. Antarmuka Aplikasi

Berikut adalah rancangan antarmuka aplikasi yang akan digunakan.

Gambar 3 22: Halaman *Login*

Gambar: 3.23. Halaman Utama

Gambar: 3.24 Halaman Penugasan

Gambar: 3.25. Halaman Buat Tugas

Gambar: 3.26 Halaman Pantau Lokasi Karyawan

3.4. Evaluasi Prototype

Setelah prototype dibangun, selanjutnya adalah melakukan evaluasi prototype apakah rancangan diterima oleh user atau tidak. Dalam penelitian ini, user yang dimaksud adalah bagian IT dan personalia. Dari hasil pertimbangan, prototype yang diajukan diterima dan dapat dilanjutkan ke proses selanjutnya.

BAB IV

IMPLEMENTASI DAN UJI COBA

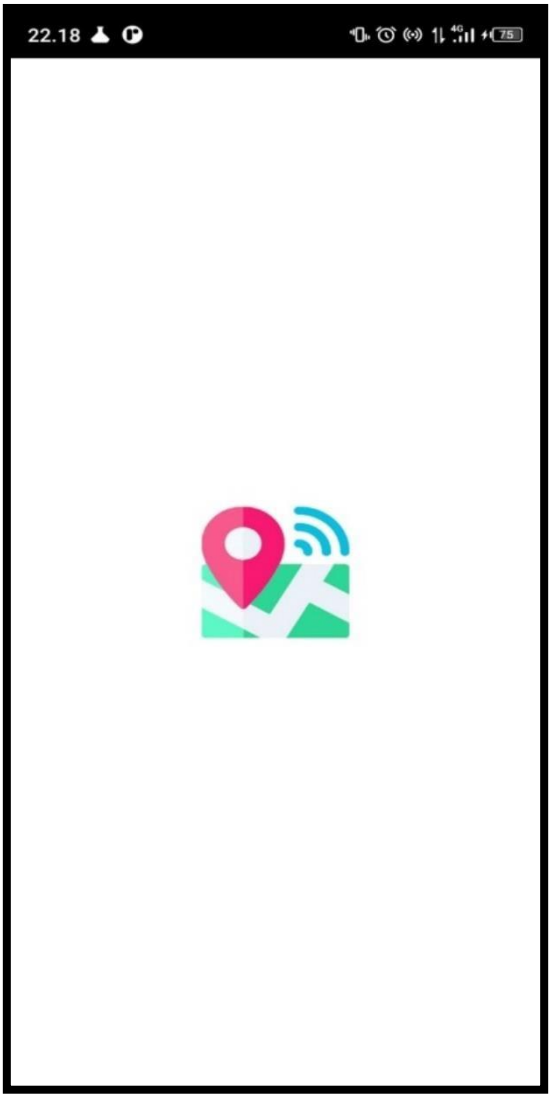
4.1. Pengkodean Sistem

Dalam tahap ini penulis berfokus pada pengkodean menggunakan bahasa pemrograman Java sebagai dasar pembuatan aplikasi atau *back end*, dan XML sebagai tampilan antarmuka (*User Interface*) atau *front end* dari aplikasi. Adapun dalam pengujian hasil akan digunakan pengujian menggunakan metode *black box testing*.

4.1.1. Implementasi

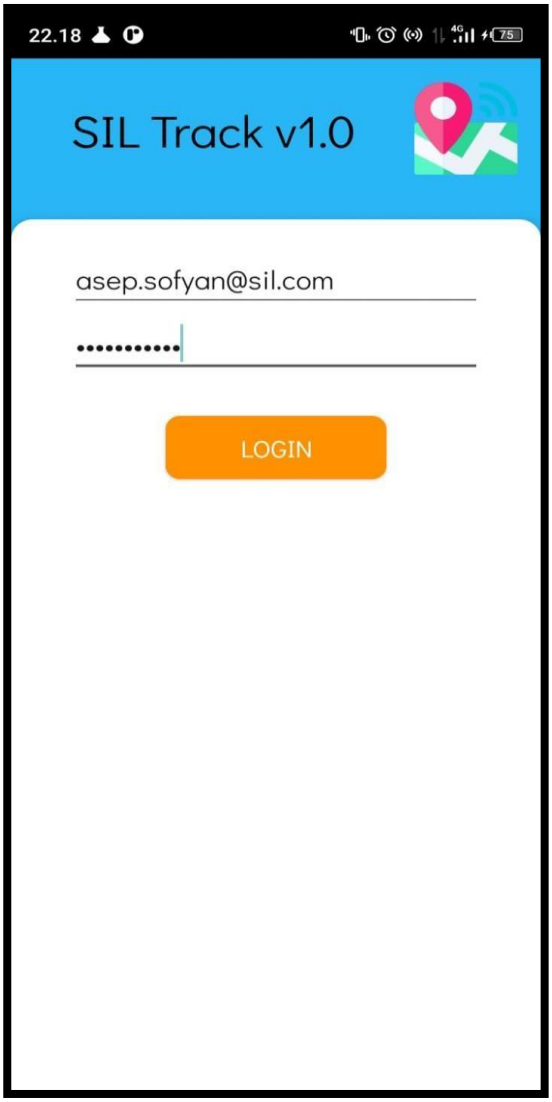
Berikut adalah halaman antarmuka bagian *end user* dari aplikasi *tracking Sales Marketing* :

1. Halaman splashscreen awal dibuka aplikasi



Gambar: 4.1. Halaman *SplashScreen*

2. Halaman saat login



Gambar: 4.2. Halaman Login

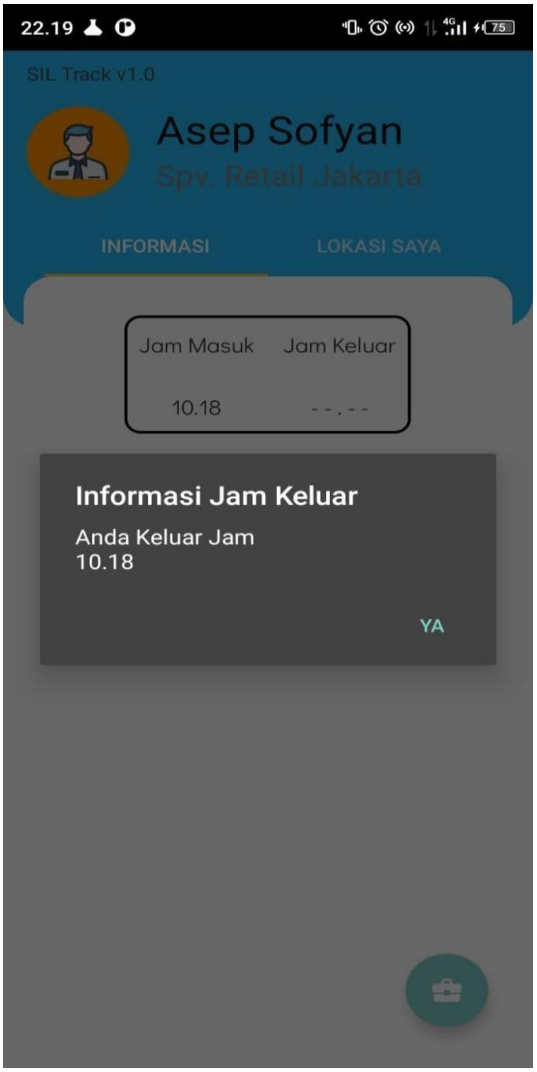
3. Halaman saat di Halaman Utama

Gambar 4.3. Halaman Utama.

4. Halaman saat mengakses jam masuk

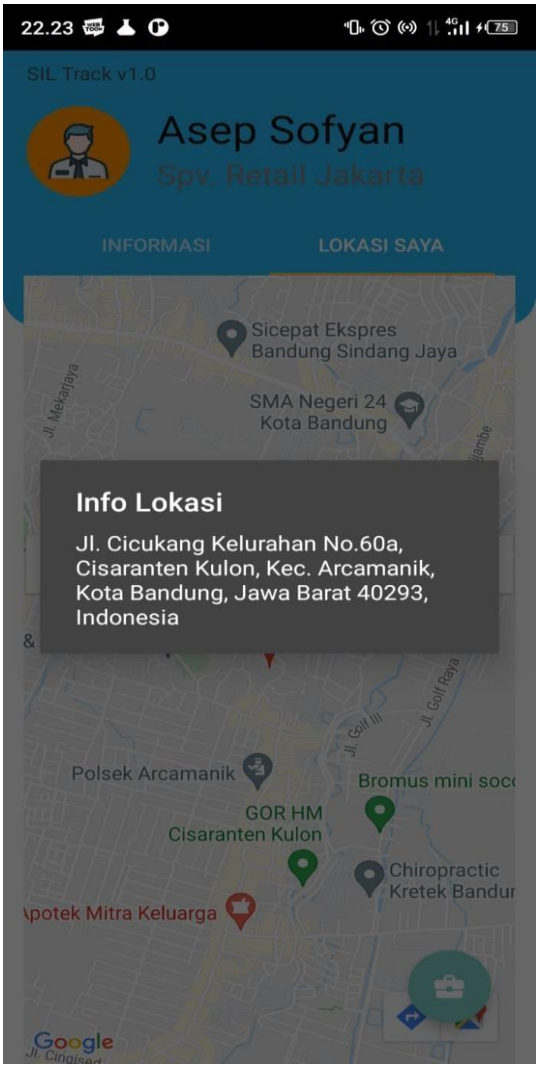
Gambar: 4.4. Halaman Konfirmasi Jam Masuk

5. Halaman saat mengakses jam keluar



Gambar: 4.5. Halaman Konfirmasi Jam Keluar

6. Halaman saat mengakses lokasi saya

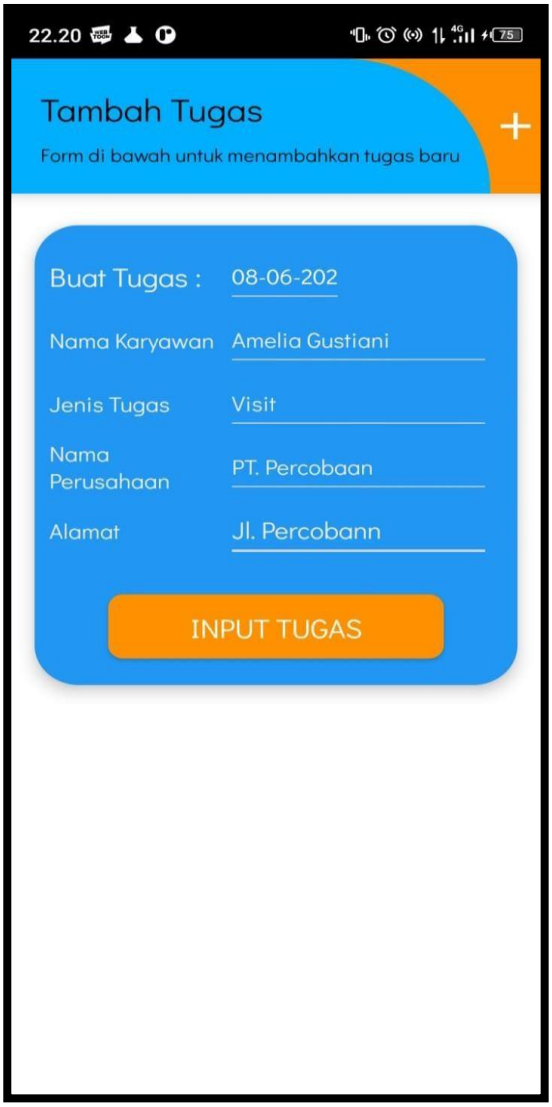


Gambar 4.6. Halaman Lokasi Saya

7. Halaman saat mengakses halaman penugasan

Gambar: 4.7. Halaman Penugasan

8. Halaman saat mengakses buat tugas

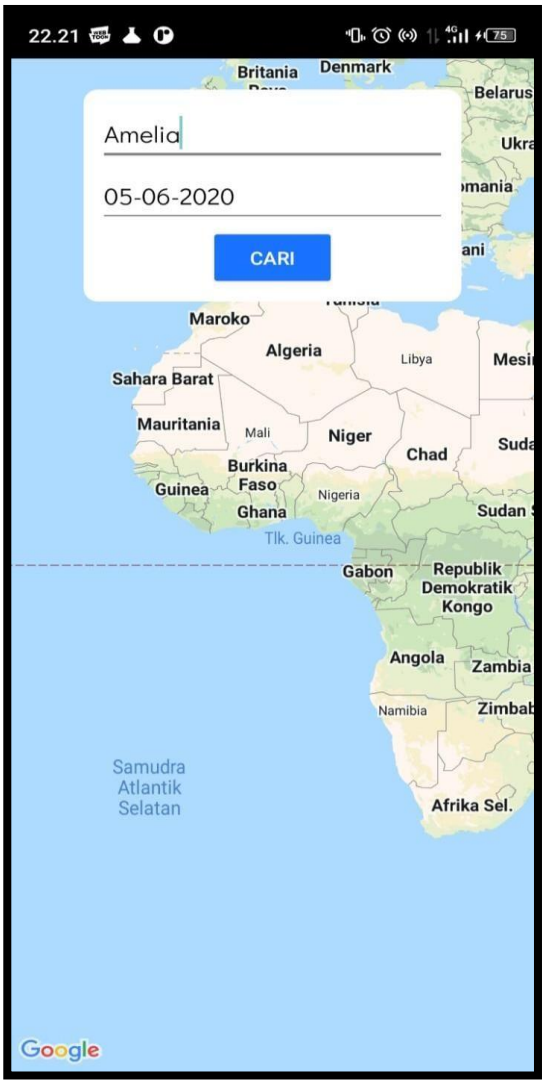


Gambar: 4.8. Halaman Buat Tugas

9. Halaman saat konfirmasi selesai buat tugas

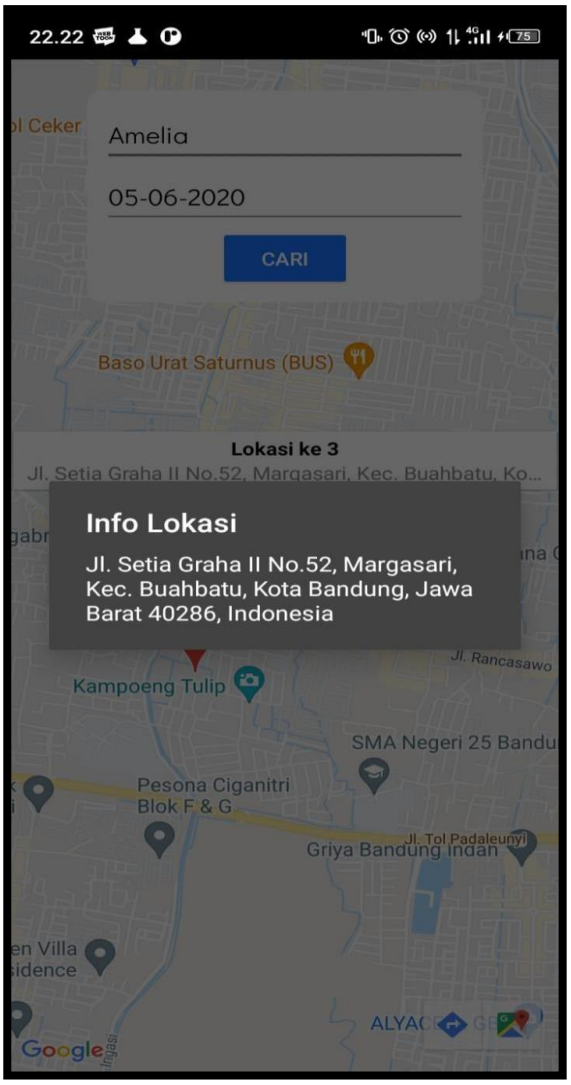
Gambar: 4.9. Halaman Konfirmasi Pembuatan Tugas

10. Halaman saat akses posisi karyawan dan mengisi form pencarian



Gambar: 4.10. Halaman Akses Pantau Karyawan

11. Halaman saat dimunculkan posisi karyawan dan detail posisi aktual



Gambar: 4.11. Halaman Detail Posisi Karyawan

4.2. Pengujian Sistem

Pengujian dilakukan dengan menjalankan aplikasi bagian *front end* dan *back end*, lalu menjalankan fitur-fitur yang telah dibuat.

4.2.1. Black Box Testing

Berikut adalah tabel hasil pengujian aplikasi menggunakan metode black box testing :

Tabel 4.1. Pengujian Black Box Pada Login

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
1.	Menguji ketika *user* tidak men-*input* alamat *email* tapi men-*input* bagian *password* saja	Alamat *email* kosong, *password* terisi	Menampilkan pesan "alamat email kosong"	Muncul pesan "alamat email kosong"	Sesuai dengan yang diharapkan
2.	Menguji ketika *user* tidak input password tapi	Password kosong, alamat email	Menampilkan pesan "password kosong"	Muncul pesan "password kosong"	Sesuai dengan yang diharapkan

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
	input bagian alamat email saja	terisi			
3.	Menguji ketika user input bagian alamat email dan password tapi tidak terdaftar di server	Alamat email terisi, password terisi	Menampilkan pesan “akun ini tidak ada di dalam list user pada server, hubungi bagian IT”	Muncul pesan “akun ini tidak ada di dalam list user pada server, hubungi bagian IT”	Sesuai dengan yang diharapkan
4.	Menguji ketika user input bagian alamat email dan password dengan akun terdaftar di server	Alamat email terisi, password terisi	Masuk ke Halaman Utama	Langsung berpindah ke halaman utama	Sesuai dengan yang diharapkan

Tabel 4.2. Pengujian Black Box Pada Halaman Utama

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
1.	Menguji ketika *user* men-klik "Jam Masuk"	Klik "Jam Masuk"	Menampilkan waktu aktual dan menyimpan lokasi aktual pada server	Muncul waktu yang nilainya sama seperti jam pada *handphone*	Sesuai dengan yang diharapkan
2.	Menguji ketika *user* men-klik "Jam Keluar"	Klik "Jam Keluar"	Menampilkan waktu aktual dan menyimpan lokasi aktual pada server	Muncul waktu yang nilainya sama seperti jam pada *handphone*, lokasi tersimpan ke dalam server	Sesuai dengan yang diharapkan
3.	Menguji ketika user men-klik tugas yang diberikan	Klik tugas	Menampilkan pop-up apakah tugas sudah selesai, dan menyimpan	Muncul pemberitahuan apakah tugas sudah selesai, lokasi terakhir tersimpan	Sesuai dengan yang diharapkan

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
			lokasi aktual pada server	pada server	
4.	Menguji ketika user men-klik tab "Lokasi saya"	Klik tab "Lokasi saya"	Menampilkan pada map posisi aktual user	Muncul *marker position* aktual *user*	Sesuai dengan yang diharapkan

Tabel 4.3. Pengujian Black Box Saat Login Dengan Level Account

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
1.	Menguji ketika *user* login dengan level akun *Sales Marketing* atau driver	Alamat email dan password	Pada Halaman Utama, tidak dimunculkan *floating button* untuk membuat tugas dan memantau posisi *Sales*	*Floating Button* tidak muncul	Sesuai dengan yang diharapkan

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
			Marketing atau driver		
2.	Menguji ketika *user* login dengan level akun kepala bagian marketing atau personalia	Alamat email dan password	Pada Halaman Utama, dimunculkan *floating button* untuk membuat tugas dan memantau posisi *Sales Marketing* atau driver	*Floating Button* muncul	Sesuai dengan yang diharapkan
3.	Menguji ketika *user* men-klik *floating button*	Klik *floating button*	Masuk ke Halaman Penugasan	Langsung berpindah ke halaman penugasan	Sesuai dengan yang diharapkan

Tabel 4.4. Pengujian Black Box Saat Pembuatan Tugas

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
1.	Menguji ketika *user* tidak input nama saat buat tugas	Nama kosong	Menampilkan pesan "nama tidak boleh kosong"	Muncul pesan "nama tidak boleh kosong"	Sesuai dengan yang diharapkan
2.	Menguji ketika *user* tidak input nama perusahaan saat buat tugas	Nama perusahaan kosong	Menampilkan pesan "nama perusahaan tidak boleh kosong"	Muncul pesan "nama perusahaan tidak boleh kosong"	Sesuai dengan yang diharapkan
3.	Menguji ketika tidak input alamat saat buat tugas	Alamat kosong	Menampilkan pesan "alamat tidak boleh kosong"	Muncul pesan "alamat tidak boleh kosong"	Sesuai dengan yang diharapkan
4.	Menguji ketika *user* input semua	Tanggal tugas terisi, nama	Menampilkan pesan "data sudah diinput	Muncul pemberitahuan "data sudah	Sesuai dengan yang

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
	isian	terisi, jenis tugas terisi, nama perusahaan terisi, alamat terisi	di server”	diinput di server”	diharapkan

Tabel 4.5. Pengujian Black Box Saat Pencarian Lokasi *Sales Marketing* / Driver

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
1.	Menguji ketika *user* tidak input nama saat cari posisi karyawan	Nama kosong	Menampilkan pesan “nama tidak boleh kosong”	Muncul pesan “nama tidak boleh kosong”	Sesuai dengan yang diharapkan
2.	Menguji ketika *user* menginput semua isian	Nama terisi, tanggal pencarian terisi	Menampilkan posisi nama yang bersangkutan pada tanggal tersebut di map	Muncul *marker position* sebanyak titik lokasi yang ada pada tanggal pencarian	Sesuai dengan yang diharapkan
3.	Menguji ketika *user* men-klik balon *windows*	Klik balon *windows information*	Menampilkan *pop-up* detail alamat	Muncul detail alamat	Sesuai dengan yang diharapkan

No	Deskripsi Pengujian	Masukan	Hasil Yang Diharapakan	Hasil Pengujian	Kesimpulan
	information pada map	pada map		pada *marker position* saat di klik	

4.3. Evaluasi sistem

Tahapan selanjutnya adalah evaluasi sistem. Pada tahapan ini sistem akan dievaluasi berdasarkan hasil pengujian sistem, jika hasil pengujian menunjukkan bahwa sistem masih terdapat kekurangan atau tidak sesuai dengan permintaan dari *user*, maka tahapan akan diulangi kembali dari pengkodean sistem, kemudian pengujian sistem kembali, sampai evaluasi sistem kembali sampai sistem sesuai dengan yang diharapkan oleh *user*.

Setelah melihat hasil pengujian sistem diatas, dapat dinilai bahwa sistem yang diusulkan sudah dapat digunakan dan sesuai dengan kebutuhan baik bagian personalia maupun kepala bagian admin. Hal ini dibuktikan dengan hasil pengujian yang menunjukkan bahwa semua pengujian menghasilkan nilai yang sesuai dengan kondisi yang diujikan.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan seluruh hasil tahapan penelitian yang telah dilakukan pada pembuatan aplikasi *tracking Sales Marketing* berbasis android menggunakan metode *reverse-geocoding* di PT. Sri Indah Labetama dapat disimpulkan bahwa memonitoring *Sales Marketing* ketika sedang melakukan *visit* ke *customer* dapat dilakukan dengan memanfaatkan penggunaan aplikasi android, yaitu dengan membuat sebuah aplikasi *tracking Sales Marketing* yang menggunakan metode *reverse-geocoding*. adapun penggunaan metode *reverse geocoding* ini dipilih karena metode ini dapat merubah nilai atau titik koordinat yang dikirim oleh *Sales Marketing* ketika sedang melakukan kunjungan atau *visit* ke *customer*, kepala bagian dapat mengetahui lokasi aktual dari *Sales Marketing* karena yang dikirim oleh *Sales Marketing* berupa titik koordinat *latitude* dan *longitude* yang tidak dimengerti oleh manusia, bukan alamat nyata yang biasa diketahui oleh manusia. selain itu, hasil pengujian juga membuktian bahwa pembuatan aplikasi ini dapat berjalan sesuai yang diharapkan, baik saat pembuatan tugas maupun saat melakukan pemantauan posisi *Sales Marketing*.

5.2. Saran

Berdasar kepada adanya keterbatasan dalam pembuatan aplikasi *tracking* ini, maka terdapat beberapa saran yang dapat menjadi bahan pertimbangan bagi pengembangan aplikasi kedepannya, yaitu :

1. Menambahkan fitur rekapan laporan kunjungan bagi kepala bagian *Sales Marketing* dan bagian personalia
2. Mengintegrasikan aplikasi dengan *database* yang sudah terpasang di perusahaan, sehingga keamanannya lebih terjamin
3. Menambahkan fitur hitung jarak dari posisi marketing saat ini dengan lokasi tujuan
4. Menambahkan fitur estimasi waktu sampai dari kantor dengan lokasi tujuan

DAFTAR PUSTAKA

Afrizal Subhan, A. (2017). Rancang Bangun Aplikasi Pembelajaran Dasar Pemrograman Berbasis Mobile Phone. Jurnal Teknik Informatika Politeknik Sekayu (TIPS), 6(1), 4–19.

Alfeno, S., & Devi, R. E. C. (2017). Implementasi Global Positioning System (GPS) dan Location Based Service (LSB) pada Sistem Informasi Kereta Api untuk Wilayah Jabodetabek. Sisfotek Global, 7(2), 27–33. https://journal.stmikglobal.ac.id/index.php/sisfotek/article/view/146

Anugrah, C. S., Santoso, H. B., & Budi, I. (2019). SISTEM INFORMASI GEOGRAFI PARIWISATA HALAL BERBASIS ANDROID DENGAN METODE GEOLOCATION (STUDI KASUS: KOTA SANTRI KABUPATEN JOMBANG). e-Prosiding SNasTekS, 1(1), 83-88.

Apa itu Java ? Inilah Ulasan Lengkapnya ! - Badoy Studio. (2022, Mei 6). Diakses 24 Mei 2022, dari website: https://badoystudio.com/apa-itu-java/

Apa itu NoSQL? Simak Penjelasan Lengkapnya! (2021, February 12). Diakses 30 Mei 2022, dari Niagahoster Blog website: https://www.niagahoster.co.id/blog/nosql-adalah/

Artikel Pro.Co.Id. (2020, September 21). Diakses 13 Maret 2022, dari website: https://www.seputarpengetahuan.co.id/2020/09/ flowchart.html

Binar Academy. (2022, Mei 12). Urutan Versi Android Dari Awal Hingga Terbaru Yang Harus Kalian Tahu. Diakses 24 Mei 24, 2022, dari Binaracademy.com website: https://www.binaracademy.com/blog/urutan-versi-android

Cloud Firestore | Simpan dan sinkronkan data aplikasi dalam skala global | Firebase. (2022). Diakses 24 Mei 2022, dari Firebase website: https://firebase.google.com/products/firestore?hl=id#:~:text=Cloud%20Firestore%20adalah%20database%20dokumen,dan%20web%20dalam%20skala%20global.

Dasar-dasar Pembuatan Layout Aplikasi Android dengan XML. (2017). Diakses

29 Mei 2022, dari https://www.codepolitan.com /dasar-dasar-pembuatanlayout-aplikasi-android-dengan-xml-599a83620ec79/

Diagram Sequence Dalam Analisa & Desain Sistem Informasi | BINUS UNIVERSITY MALANG | Pilihan Universitas Terbaik di Malang. (2020, Desember 15). Diakses 26 Mei 2022, dari BINUS UNIVERSITY MALANG | Pilihan Universitas Terbaik di Malang website: https://binus.ac.id/malang/2020/12/diagram-sequence-dalam-analisa-desain-sistem-informasi/

Entity Relationship Diagram (ERD) - What is an ER Diagram? (2022). Diakses 15 Juni 2022, dari : https://www.smartdraw.com/entity-relationship-diagram/

Evan, A., Santoso, L. W., & Adipranata, R. (2020). Sistem Pembacaan Kode Pos yang Terintegrasi dengan Pencarian Alamat pada OpenStreetMap. Jurnal Infra, 8(1), 53-58.

Fauzi, R. (2021). Bab 2 tinjauan pustaka. 8–45.

Galih Pradana, A., & Nita, S. (2019). Rancang Bangun Game Edukasi "AMUDRA" Alat Musik Daerah Berbasis Android. Jurnal Seminar Nasional Teknologi Informasi Dan Komunikasi 2019, 2(1), 49–53.

Kurniawan, I. D. (2014). Aplikasi Monitoring Keberadaan Objek Melalui Perangkat Bergerak Berbasis Android (Doctoral dissertation, Institut Teknologi Sepuluh Nopember).

Management Information System - Universitas Surabaya. (2014). Android: Sistem Operasi Pada *Smartphone* | Universitas Surabaya (UBAYA). Diakses 24 Mei 2022 dari https://ubaya.ac.id/2014/content/articles_detail/7/Android--Sistem-Operasi-pada-*Smartphone*.html

Mengenal Android Studio | Developer Android | Android Developers. (2021). Diakses 29 Mei 2022 dari https://developer.android.com/studio/intro?hl=id52

Panduan Lengkap XML: Pengertian, Contoh, dan Cara Membuka Filenya. (2020, Juli 17). Diakses 29 Mei 2022, dari https://www.niagahoster.co.id/blog/xml/#Apa_itu_XML_dan_Manfaatnya

Purnomo, D. (2017). Model Prototyping Pada Pengembangan Sistem Informasi. J I M P - Jurnal Informatika Merdeka Pasuruan, 2(2), 54–61. https://doi.org/10.37438/jimp.v2i2.67

Rati, R. A., Muda, O. C., & Pernando, S. L. Analisis Dan Desain Penerapan Location Based Service (LBS) Dan Geographic Information System (GIS) Pada Aplikasi Pemesanan Jasa Laundry.

Redaksi Jagoan Hosting. (2022, February 10). 7 Contoh Use Case Diagram, Simbol hingga Cara Buatnya. Diakses 15 Juni 2022, dari https://www.jagoanhosting.com/blog/use-case-diagram/

Riyono, B., & Rifkianti, W. (2018). Android-Based Information System Of Online Teaching Services With Geo-Location Determination. bit-Tech, 1(1), 9-18.

Rosaly, R., & Prasetyo, A. (2019). Pengertian Flowchart Beserta Fungsi dan Simbol-simbol Flowchart yang Paling Umum Digunakan. Https://Www.Nesabamedia.Com, 2, 2. https://www.nesabamedia.com/pengertian-flowchart/https://www.nesabamedia.com/pengertian-flowchart/

Suliyanti, W. N. (2019). Studi Literatur Basis Data SQL dan NoSQL. Kilat, 8(1), 48–51. https://doi.org/10.33322/kilat.v8i1.460

Suratno, S. (2019). PENERAPAN REVERSE GEOCODING UNTUK APLIKASI ABSENSI MOBILE DI PT. KENCANA ALAM PUTRA (Doctoral dissertation, Politeknik NSC Surabaya).

Tata letak | Developer Android | Android Developers. (2021). Diakses 29 Mei 2022 dari https://developer.android.com/guide/topics/ui/declaring-layout?hl=id

UML Diagram : Activity Diagram. (2019, November 22). Diakses 26 Mei 2022, dari https://socs.binus.ac.id/2019/11/22/uml-diagram-activity-diagram/

Use Case Diagram: Pengertian, Fungsi, Teknik, Dan Contoh. (2021, Juni 21). Diakses 22 Mei 2022, from https://www.sekawanmedia.co.id/blog/use-case-diagram/

LAMPIRAN

Lampiran 1

Source Code Java

account_client.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.app.Application;
4.
5. public class account_client extends Application {
6.     private default_account default_account = null;
7.
8.     public default_account getdefault_account() {
9.         return default_account;
10.    }
11.
12.    public void setdefault_account(default_account
   default_account) {
13.        this.default_account = default_account;
14.    }
15. }
```

default_account.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.os.Parcel;
4. import android.os.Parcelable;
5.
6. import com.google.firebase.firestore.GeoPoint;
7. import com.google.firebase.firestore.ServerTimestamp;
8.
9. import java.util.Date;
10.
11.public class default_account implements Parcelable {
12.
13.     private String email;
14.     private String passwd;
15.     private String jabatan;
16.     private GeoPoint geoPoint;
17.     private @ServerTimestamp
18.     Date timestamp;
19.
20.
21.     public default_account() {
22.     }
23.
24.public default_account(String email, String passwd, String
   jabatan, GeoPoint geoPoint, Date timestamp) {
25.         this.email = email;
26.         this.passwd = passwd;
27.         this.jabatan = jabatan;
28.         this.geoPoint = geoPoint;
29.         this.timestamp = timestamp;
30.     }
31.
32.     public String getJabatan() {
33.         return jabatan;
34.     }
35.
36.     public void setJabatan(String jabatan) {
37.         this.jabatan = jabatan;
38.     }
39.
40.     public Date getTimestamp() {
41.         return timestamp;
42.     }
43.
44.     public void setTimestamp(Date timestamp) {
45.         this.timestamp = timestamp;
46.     }
47.
48.     public GeoPoint getGeoPoint() {
```

```
49.         return geoPoint;
50.     }
51.
52.     public void setGeoPoint(GeoPoint geoPoint) {
53.         this.geoPoint = geoPoint;
54.     }
55.
56.     protected default_account(Parcel in) {
57.         email = in.readString();
58.         passwd = in.readString();
59.         jabatan = in.readString();
60.     }
61.
62. public static final Creator<default_account> CREATOR = new
    Creator<default_account>() {
63.         @Override
64.         public default_account createFromParcel(Parcel in) {
65.
66.             return new default_account(in);
67.         }
68.
69.         @Override
70.         public default_account[] newArray(int size) {
71.
72.             return new default_account[size];
73.         }
74.     };
75.
76.     public String getEmail() {
77.
78.         return email;
79.     }
80.
81.     public void setEmail(String email) {
82.
83.         this.email = email;
84.     }
85.
86.     public String getPasswd() {
87.
88.         return passwd;
89.     }
90.
91.     public void setPasswd(String passwd) {
92.
93.         this.passwd = passwd;
94.     }
95.
96.
97.     @Override
98.     public int describeContents() {
99.         return 0;
```

```
100.     }
101.
102.     @Override
103.     public void writeToParcel(Parcel dest, int flags) {
104.         dest.writeString(email);
105.         dest.writeString(passwd);
106.         dest.writeString(jabatan);
107.     }
108. }
```

file_tugas.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. public class file_tugas {
4.     String jenis_tugas;
5.     String alamat_tujuan;
6.     String nama_perusahaan;
7.
8.     public file_tugas() {
9.     }
10.
11.    public file_tugas(String jenis_tugas, String alamat_tujuan,
   String nama_perusahaan) {
12.        this.jenis_tugas = jenis_tugas;
13.        this.alamat_tujuan = alamat_tujuan;
14.        this.nama_perusahaan = nama_perusahaan;
15.    }
16.
17.    public String getjenis_tugas() {
18.        return jenis_tugas;
19.    }
20.
21.    public void setjenis_tugas(String jenis_tugas) {
22.        this.jenis_tugas = jenis_tugas;
23.    }
24.
25.    public String getalamat_tujuan() {
26.        return alamat_tujuan;
27.    }
28.
29.    public void setalamat_tujuan(String alamat_tujuan) {
30.        this.alamat_tujuan = alamat_tujuan;
31.    }
32.
33.    public String getnama_perusahaan() {
34.        return nama_perusahaan;
35.    }
36.
37.    public void setnama_perusahaan(String nama_perusahaan) {
38.        this.nama_perusahaan = nama_perusahaan;
39.    }
40. }
```

fragment_daftar_tugas.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.Manifest;
4. import android.app.AlertDialog;
5. import android.content.DialogInterface;
6. import android.content.pm.PackageManager;
7. import android.location.Location;
8. import android.os.Build;
9. import android.os.Bundle;
10. import android.util.ArrayMap;
11. import android.view.LayoutInflater;
12. import android.view.View;
13. import android.view.ViewGroup;
14. import android.widget.ImageView;
15. import android.widget.TextView;
16. import android.widget.Toast;
17.
18. import androidx.annotation.NonNull;
19. import androidx.annotation.Nullable;
20. import androidx.annotation.RequiresApi;
21. import androidx.core.app.ActivityCompat;
22. import androidx.fragment.app.Fragment;
23. import androidx.recyclerview.widget.ItemTouchHelper;
24. import androidx.recyclerview.widget.LinearLayoutManager;
25. import androidx.recyclerview.widget.RecyclerView;
26.
27. import com.firebase.ui.firestore.FirestoreRecyclerOptions;
28. import
    com.google.android.gms.location.FusedLocationProviderClient;
29. import com.google.android.gms.location.LocationServices;
30. import com.google.android.gms.tasks.OnCompleteListener;
31. import com.google.android.gms.tasks.Task;
32. import com.google.firebase.auth.FirebaseAuth;
33. import com.google.firebase.auth.FirebaseUser;
34. import com.google.firebase.firestore.CollectionReference;
35. import com.google.firebase.firestore.DocumentReference;
36. import com.google.firebase.firestore.DocumentSnapshot;
37. import com.google.firebase.firestore.EventListener;
38. import com.google.firebase.firestore.FieldValue;
39. import com.google.firebase.firestore.FirebaseFirestore;
40. import com.google.firebase.firestore.FirebaseFirestoreException;
41. import com.google.firebase.firestore.GeoPoint;
42. import com.google.firebase.firestore.Query;
43. import com.google.firebase.firestore.QuerySnapshot;
44. import com.google.firebase.firestore.model.DocumentCollections;
45.
46. import java.text.SimpleDateFormat;
47. import java.util.ArrayList;
48. import java.util.Arrays;
```

```
49. import java.util.Date;
50. import java.util.HashMap;
51. import java.util.List;
52. import java.util.Map;
53.
54. public class fragment_daftar_tugas extends Fragment {
55.     private RecyclerView recyclerView;
56. private FirebaseFirestore firestore =
    FirebaseFirestore.getInstance();
57.     private FirebaseAuth firebaseAuth;
58.     private FirebaseAuth.AuthStateListener stateListener;
59. private CollectionReference reference =
    firestore.collection("id_task");
60.     private DocumentReference documentReference;
61.     private tugas_adapter adapter;
62.     private ImageView img;
63. private TextView notif_selesai, tv_judul_masuk,
    tv_judul_keluar, masuk, keluar;
64.     private FusedLocationProviderClient client;
65.     public String NAMA_KARYAWAN = "";
66.
67.     @Override
68. public View onCreateView(LayoutInflater inflater, ViewGroup
    container,
69.                              Bundle savedInstanceState) {
70.         // Inflate the layout for this fragment
71.
72.         setup_user();
73. View view =
    inflater.inflate(R.layout.fragment_daftar_tugas, container,
    false);
74. client =
    LocationServices.getFusedLocationProviderClient(getActivity());
75.         Date jam = new Date();
76.         SimpleDateFormat format = new SimpleDateFormat("hh.mm");
77.         tv_judul_keluar = view.findViewById(R.id.judul_keluar);
78.         tv_judul_masuk = view.findViewById(R.id.judul_masuk);
79.         masuk = view.findViewById(R.id.tv_masuk);
80.         keluar = view.findViewById(R.id.tv_keluar);
81. tv_judul_masuk.setOnClickListener(new
    View.OnClickListener() {
82.             @Override
83.             public void onClick(View v) {
84. AlertDialog.Builder builder = new
    AlertDialog.Builder(view.getContext());
85.                 builder.setTitle("Informasi Jam Masuk")
86. .setMessage("Anda Masuk Jam \n" +format.format(jam))
87. .setPositiveButton("Ya", new
    DialogInterface.OnClickListener() {
88.                             @Override
```

```
89. public void onClick(DialogInterfacedialog, int which) {
90.
    masuk.setText(format.format(jam));
91.                                 if
    (ActivityCompat.checkSelfPermission(getActivity(),
92.
    Manifest.permission.ACCESS_FINE_LOCATION) ==
    PackageManager.PERMISSION_GRANTED) {
93.                                     submit_location();
94.                                 } else {
95.
    ActivityCompat.requestPermissions(getActivity(),
96.                                                 new
    String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 44);
97.                                 }
98.
99.                             }
100.                        });
101.                AlertDialog pop_up = builder.create();
102.                pop_up.show();
103.
104.            }
105.        });
106.        tv_judul_keluar.setOnClickListener(new
    View.OnClickListener() {
107.            @Override
108.            public void onClick(View v) {
109.                AlertDialog.Builder builder = new
    AlertDialog.Builder(view.getContext());
110.                builder.setTitle("Informasi Jam Keluar")
111.                        .setMessage("Anda Keluar Jam \n" +
    format.format(jam))
112.                        .setPositiveButton("Ya", new
    DialogInterface.OnClickListener() {
113.                            @Override
114.                            public void onClick(DialogInterface
    dialog, int which) {
115.
    keluar.setText(format.format(jam));
116.                                dialog.cancel();
117.                            }
118.                        });
119.                AlertDialog pop_up = builder.create();
120.                pop_up.show();
121.            }
122.        });
123.
124.        Query query = reference.orderBy("jenis_tugas",
    Query.Direction.ASCENDING);//.whereEqualTo("nama_karyawan",
    "Amelia Gustiani");
125.        //orderBy("judul_tugas", Query.Direction.DESCENDING).;
```

```
126.
127.         query.addSnapshotListener(new
   EventListener<QuerySnapshot>() {
128.             @Override
129.             public void onEvent(@Nullable QuerySnapshot value,
   @Nullable FirebaseFirestoreException error) {
130.                 if (value.isEmpty()) {
131.                     img.setVisibility(View.VISIBLE);
132.                     notif_selesai.setVisibility(View.VISIBLE);
133.                     notif_selesai.setText("Hore, Tidak ada
   tugas untuk kamu ! ");
134.                 }
135.             }
136.         });
137.
138.         FirestoreRecyclerOptions<file_tugas> options = new
   FirestoreRecyclerOptions.Builder<file_tugas>()
139.                 .setQuery(query, file_tugas.class)
140.                 .build();
141.
142.         img = (ImageView) view.findViewById(R.id.img_selesai);
143.         img.setVisibility(View.INVISIBLE);
144.         notif_selesai = (TextView)
   view.findViewById(R.id.textView29);
145.         notif_selesai.setVisibility(View.INVISIBLE);
146.         adapter = new tugas_adapter(options);
147.
148.         recyclerView = view.findViewById(R.id.rv_daftar_tugas);
149.         recyclerView.setLayoutManager(new
   LinearLayoutManager(view.getContext()));
150.         recyclerView.setHasFixedSize(true);
151.         recyclerView.setAdapter(adapter);
152.
153.         new ItemTouchHelper(new
   ItemTouchHelper.SimpleCallback(0,
154.                 ItemTouchHelper.LEFT | ItemTouchHelper.RIGHT) {
155.             @Override
156.             public boolean onMove(@NonNull RecyclerView
   recyclerView, @NonNull RecyclerView.ViewHolder viewHolder,
   @NonNull RecyclerView.ViewHolder target) {
157.                 return false;
158.             }
159.
160.             @Override
161.             public void onSwiped(@NonNull
   RecyclerView.ViewHolder viewHolder, int direction) {
162.                 AlertDialog.Builder prompt = new
   AlertDialog.Builder(view.getContext());
163.                 prompt.setTitle("Konfirmasi selesai tugas ?")
164.                         .setMessage("Lokasi anda akan disimpan
   di database")
```

```
165.                         .setPositiveButton("Ya", new
    DialogInterface.OnClickListener() {
166.                             @Override
167.                             public void onClick(DialogInterface
    dialog, int which) {
168.
    adapter.konfirm_tugas(viewHolder.getAdapterPosition());
169.                             }
170.                         })
171.                         .setNegativeButton("Tidak", new
    DialogInterface.OnClickListener() {
172.                             @Override
173.                             public void onClick(DialogInterface
    dialog, int which) {
174.                                 dialog.cancel();
175.                             }
176.                         });
177.                 AlertDialog dialog = prompt.create();
178.                 dialog.show();
179.             }
180.         }).attachToRecyclerView(recyclerView);
181.
182.         adapter.setOnClickListener(new
    tugas_adapter.OnItemClickListener() {
183.             @Override
184.             public void OnItemClick(DocumentSnapshot
    documentSnapshot, int position) {
185.                 try {
186.                     update_location_task();
187.                 } catch (Exception e){
188.                     Toast.makeText(view.getContext(), ""+e,
    Toast.LENGTH_SHORT).show();
189.                 }
190.             }
191.         });
192.
193.         return view;
194.     }
195.
196.     private void setup_user() {
197.         firebaseAuth = FirebaseAuth.getInstance();
198.         stateListener = new FirebaseAuth.AuthStateListener() {
199.             @Override
200.             public void onAuthStateChanged(@NonNull
    FirebaseAuth firebaseAuth) {
201.
202.                 FirebaseUser user =
    firebaseAuth.getCurrentUser();
203.
204.                 if (user != null) {
205.                     firestore =
    FirebaseFirestore.getInstance();
```

```
206.                     documentReference =
    firestore.collection("data_user").document(user.getUid());
207.
    documentReference.get().addOnCompleteListener(new
    OnCompleteListener<DocumentSnapshot>() {
208.                         @Override
209.                         public void onComplete(@NonNull
    Task<DocumentSnapshot> task) {
210.                             try {
211.                                 String karyawan =
    task.getResult().getString("nama");
212.                                 String bagian =
    task.getResult().getString("bagian");
213.                                 NAMA_KARYAWAN = karyawan;
214.
215.                             } catch (Exception e) {
216.                                 Toast.makeText(getActivity(),
    "" + e, Toast.LENGTH_SHORT).show();
217.                             }
218.                         }
219.                     });
220.                 }
221.             }
222.         };
223.     }
224.
225.     private void submit_location() {
226.         if (ActivityCompat.checkSelfPermission(getActivity(),
    Manifest.permission.ACCESS_FINE_LOCATION) !=
    PackageManager.PERMISSION_GRANTED &&
    ActivityCompat.checkSelfPermission(getActivity(),
    Manifest.permission.ACCESS_COARSE_LOCATION) !=
    PackageManager.PERMISSION_GRANTED) {
227.             // TODO: Consider calling
228.             //    ActivityCompat#requestPermissions
229.             // here to request the missing permissions, and
    then overriding
230.             //   public void onRequestPermissionsResult(int
    requestCode, String[] permissions,
231.             //                                          int[]
    grantResults)
232.             // to handle the case where the user grants the
    permission. See the documentation
233.             // for ActivityCompat#requestPermissions for more
    details.
234.             return;
235.         }
236.         client.getLastLocation().addOnCompleteListener(new
    OnCompleteListener<Location>() {
237.
238.             @Override
```

```
239.             public void onComplete(@NonNull Task<Location>
    task) {
240.                 Location location = task.getResult();
241.                 if (location != null) {
242.                     double latitude = location.getLatitude();
243.                     double longitude = location.getLongitude();
244.                     Date date = new Date();
245.
246.                     SimpleDateFormat format = new
    SimpleDateFormat("dd-MM-yyyy");
247.                     String jam = "";
248.                     jam = String.valueOf(format.format(date));
249.                     GeoPoint geoPoint = new GeoPoint(latitude,
    longitude);
250.
251.                     Map<String, Object> objectMap = new
    HashMap<>();
252.                     objectMap.put(jam,
    Arrays.asList(geoPoint));
253.                     objectMap.put("nama", NAMA_KARYAWAN);
254.                     DocumentReference document =
    firestore.collection("data_location").document(firebaseAuth.getUi
    d());
255.
    document.set(objectMap).addOnCompleteListener(new
    OnCompleteListener<Void>() {
256.                         @Override
257.                         public void onComplete(@NonNull
    Task<Void> task) {
258.                             if (task.isSuccessful()) {
259.                                 Toast.makeText(getActivity(),
    "Berhasil Update", Toast.LENGTH_SHORT).show();
260.                             }
261.                         }
262.                     });
263.                 }
264.             }
265.         });
266.     }
267.
268.     private void update_location_task() {
269.         if (ActivityCompat.checkSelfPermission(getActivity(),
    Manifest.permission.ACCESS_FINE_LOCATION) !=
    PackageManager.PERMISSION_GRANTED &&
    ActivityCompat.checkSelfPermission(getActivity(),
    Manifest.permission.ACCESS_COARSE_LOCATION) !=
    PackageManager.PERMISSION_GRANTED) {
270.             // TODO: Consider calling
271.             //    ActivityCompat#requestPermissions
272.             // here to request the missing permissions, and
    then overriding
```

```
273.             //   public void onRequestPermissionsResult(int
   requestCode, String[] permissions,
274.             //                                          int[]
   grantResults)
275.             // to handle the case where the user grants the
   permission. See the documentation
276.             // for ActivityCompat#requestPermissions for more
   details.
277.             return;
278.         }
279.         client.getLastLocation().addOnCompleteListener(new
   OnCompleteListener<Location>() {
280.
281.             @Override
282.             public void onComplete(@NonNull Task<Location>
   task) {
283.                 Location location = task.getResult();
284.                 if (location != null) {
285.                     double latitude = location.getLatitude();
286.                     double longitude = location.getLongitude();
287.                     Date date = new Date();
288.
289.                     SimpleDateFormat format = new
   SimpleDateFormat("dd-MM-yyyy");
290.                     String jam = "";
291.                     jam = String.valueOf(format.format(date));
292.                     GeoPoint geoPoint = new GeoPoint(latitude,
   longitude);
293. CollectionReference collection =
   firestore.collection("data_location");
294.                     DocumentReference document =
   firestore.collection("data_location").document(firebaseAuth.getUi
   d());
295.
296.                     Map<String, Object> objectMap = new
   HashMap<>();
297.                     objectMap.put(jam,
   Arrays.asList(geoPoint));
298.                     objectMap.put("nama", NAMA_KARYAWAN);
299.                     Map<String, Object> contoh = new
   HashMap<>();
300.                     contoh.put("nama", NAMA_KARYAWAN);
301.
302.                     //document.set(objectMap);
303.
304.                     document.update(jam,
   FieldValue.arrayUnion(geoPoint)).addOnCompleteListener(new
   OnCompleteListener<Void>() {
305.                         @Override
306.                         public void onComplete(@NonNull
   Task<Void> task) {
307.                             if (task.isSuccessful()){
```

```
                                    Toast.makeText(getActivity(),
    "Berhasil ditambahkan", Toast.LENGTH_SHORT).show();
                                }
                            }
                        });
                    }
                }
            });
        }

        @Override
        public void onStart() {
            super.onStart();
            adapter.startListening();
            firebaseAuth.addAuthStateListener(stateListener);
        }

        @Override
        public void onStop() {
            super.onStop();
            adapter.stopListening();
            if (firebaseAuth != null) {
    firebaseAuth.removeAuthStateListener(stateListener);
            }
        }
    }
```

fragment_map_user.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.Manifest;
4. import android.app.Activity;
5. import android.content.pm.PackageManager;
6. import android.location.Address;
7. import android.location.Geocoder;
8. import android.location.Location;
9. import android.os.Bundle;
10. import android.view.LayoutInflater;
11. import android.view.View;
12. import android.view.ViewGroup;
13. import android.widget.Toast;
14.
15. import androidx.annotation.NonNull;
16. import androidx.appcompat.app.AlertDialog;
17. import androidx.core.app.ActivityCompat;
18. import androidx.fragment.app.Fragment;
19.
20. import
    com.google.android.gms.location.FusedLocationProviderClient;
21. import com.google.android.gms.location.LocationServices;
22. import com.google.android.gms.maps.CameraUpdateFactory;
23. import com.google.android.gms.maps.GoogleMap;
24. import com.google.android.gms.maps.OnMapReadyCallback;
25. import com.google.android.gms.maps.SupportMapFragment;
26. import com.google.android.gms.maps.model.LatLng;
27. import com.google.android.gms.maps.model.Marker;
28. import com.google.android.gms.maps.model.MarkerOptions;
29. import com.google.android.gms.tasks.OnSuccessListener;
30. import com.google.android.gms.tasks.Task;
31.
32. import java.io.IOException;
33. import java.util.List;
34. import java.util.Locale;
35.
36. public class fragment_map_user extends Fragment {
37.     private SupportMapFragment supportMapFragment;
38.     private FusedLocationProviderClient providerClient;
39.
40.
41.     @Override
42. public View onCreateView(LayoutInflater inflater, ViewGroup
    container,
43.                               Bundle savedInstanceState) {
44.         // Inflate the layout for this fragment
45.
46.         Bundle mapViewBundle = null;
47.
```

```
48. View view = inflater.inflate(R.layout.fragment_map_user,
    container, false);
49. supportMapFragment = (SupportMapFragment)
    getChildFragmentManager().findFragmentById(R.id.map);
50. providerClient =
    LocationServices.getFusedLocationProviderClient(getActivity());
51.         if (ActivityCompat.checkSelfPermission(getActivity(),
52. Manifest.permission.ACCESS_FINE_LOCATION) ==
    PackageManager.PERMISSION_GRANTED) {
53.             lokasi_saya();
54.         } else {
55.             ActivityCompat.requestPermissions(getActivity(),
56.                     new
    String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 44);
57.         }
58.
59.         return view;
60.     }
61.
62.     private void lokasi_saya() {
63. if (ActivityCompat.checkSelfPermission(getActivity(),
    Manifest.permission.ACCESS_FINE_LOCATION) !=
    PackageManager.PERMISSION_GRANTED &&
    ActivityCompat.checkSelfPermission(getActivity(),
    Manifest.permission.ACCESS_COARSE_LOCATION) !=
    PackageManager.PERMISSION_GRANTED) {
64.             // TODO: Consider calling
65.             //    ActivityCompat#requestPermissions
66. // here to request the missing permissions, and thenoverriding
67. //          public void onRequestPermissionsResult(int
    requestCode, String[] permissions,
68.             //                                          int[]
    grantResults)
69. // to handle the case where the user grants thepermission.
    See the documentation
70. // for ActivityCompat#requestPermissions for moredetails.
71.             return;
72.         }
73.         Task<Location> task = providerClient.getLastLocation();
74. task.addOnSuccessListener(new
    OnSuccessListener<Location>() {
75.             @Override
76.             public void onSuccess(Location location) {
77.                 if (location != null) {
78. supportMapFragment.getMapAsync(newOnMapReadyCallback()
    {
79.                         @Override
80. public void onMapReady(@NonNull GoogleMapgoogleMap) {
```

```
81.                             Geocoder geocoder = new
    Geocoder(getActivity(), Locale.getDefault());
82.                             double latitude =
    location.getLatitude();
83.                             double longitude =
    location.getLongitude();
84.                             List<Address> addresses = null;
85.                             try {
86.                                 addresses =
    geocoder.getFromLocation(latitude, longitude, 1);
87.                             } catch (IOException e) {
88.                                 e.printStackTrace();
89.                             }
90.
91.                             String tempat =
    addresses.get(0).getAddressLine(0);
92.                             LatLng latLng = new
    LatLng(location.getLatitude(),location.getLongitude());
93.                             googleMap.addMarker(new
    MarkerOptions().position(latLng).title("Lokasi saya saat
    ini").snippet(tempat));
94.
    googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng,
    15));
95.
96.
    googleMap.setOnInfoWindowClickListener(new
    GoogleMap.OnInfoWindowClickListener() {
97.                                 @Override
98.                                 public void
    onInfoWindowClick(@NonNull Marker marker) {
99.                                     AlertDialog.Builder builder =
    new AlertDialog.Builder(getActivity());
100.                                    builder.setTitle("Info
    Lokasi")
101.                                            .setMessage(marker.
    getSnippet());
102.                                    AlertDialog alertDialog =
    builder.create();
103.                                    alertDialog.show();
104.                                }
105.                            });
106.                        }
107.                    });
108.                } else {
109.                    Toast.makeText(getActivity(), "Lokasi gagal
    didapat", Toast.LENGTH_SHORT).show();
110.                }
111.            }
112.        });
113.    }
114. }
```

Login.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.content.Intent;
4. import android.os.Bundle;
5. import android.view.View;
6. import android.widget.Button;
7. import android.widget.EditText;
8. import android.widget.ProgressBar;
9. import android.widget.Spinner;
10. import android.widget.Toast;
11.
12. import androidx.annotation.NonNull;
13. import androidx.appcompat.app.AppCompatActivity;
14.
15. import com.google.android.gms.tasks.OnCompleteListener;
16. import com.google.android.gms.tasks.OnFailureListener;
17. import com.google.android.gms.tasks.OnSuccessListener;
18. import com.google.android.gms.tasks.Task;
19. import com.google.firebase.auth.AuthResult;
20. import com.google.firebase.auth.FirebaseAuth;
21. import com.google.firebase.auth.FirebaseUser;
22. import com.google.firebase.firestore.DocumentReference;
23. import com.google.firebase.firestore.DocumentSnapshot;
24. import com.google.firebase.firestore.FirebaseFirestore;
25.
26. import java.util.HashMap;
27. import java.util.Map;
28.
29. public class Login extends AppCompatActivity {
30.
31.     private EditText email, passwd;
32.     private Button tbl_login;
33.     private ProgressBar loading;
34.     private FirebaseAuth.AuthStateListener authStateListener;
35.     private FirebaseAuth firebaseAuth;
36.     private FirebaseFirestore firestore;
37.
38.     @Override
39.     protected void onCreate(Bundle savedInstanceState) {
40.         super.onCreate(savedInstanceState);
41.         setContentView(R.layout.activity_login);
42.
43.         email = findViewById(R.id.et_email);
44.         passwd = findViewById(R.id.et_passwd);
45.         loading = findViewById(R.id.progbar_loading);
46.         tbl_login = findViewById(R.id.btn_login);
47.         firestore = FirebaseFirestore.getInstance();
48.
49.         setupFirebase();
```

```
50.
51.         firebaseAuth = FirebaseAuth.getInstance();
52.
53.         loading.setVisibility(View.INVISIBLE);
54.
55.         tbl_login.setOnClickListener(new View.OnClickListener() {
56.             @Override
57.             public void onClick(View v) {
58.                 proses_login();
59.             }
60.         });
61.     }
62.
63.     private void proses_login() {
64.         loading.setVisibility(View.VISIBLE);
65.         String alamat_email = email.getText().toString();
66.         String pwd = passwd.getText().toString();
67.         firebaseAuth.signInWithEmailAndPassword(alamat_email,
    pwd)
68.                 .addOnCompleteListener(new
    OnCompleteListener<AuthResult>() {
69.                     @Override
70. public void onComplete(@NonNullTask<AuthResult>
    task) {
71.                         loading.setVisibility(View.INVISIBLE);
72.                         if (task.isSuccessful()) {
73. Toast.makeText(Login.this, "LoginBerhasil",
    Toast.LENGTH_SHORT).show();
74.                         } else {
75. Toast.makeText(Login.this, "LoginGagal",
    Toast.LENGTH_SHORT).show();
76.                         }
77.                     }
78.                 }).addOnFailureListener(new OnFailureListener() {
79.             @Override
80.             public void onFailure(@NonNull Exception e) {
81. Toast.makeText(Login.this, "Something Wrong",
    Toast.LENGTH_SHORT).show();
82.                 loading.setVisibility(View.INVISIBLE);
83.             }
84.         });
85.     }
86.
87.     private void setupFirebase() {
88.         authStateListener = new FirebaseAuth.AuthStateListener()
    {
89.             @Override
90. public void onAuthStateChanged(@NonNull FirebaseAuth
    firebaseAuth) {
91. FirebaseUser firebaseUser =
    firebaseAuth.getCurrentUser();
92.
```

```
                if (firebaseUser != null) {
Toast.makeText(Login.this, "Berhasil login",
Toast.LENGTH_SHORT).show();
FirebaseFirestore firestore =
FirebaseFirestore.getInstance();
DocumentReference documentReference =
firestore.collection("data_user").document(firebaseUser.getUid())
;

documentReference.get().addOnCompleteListener(new
OnCompleteListener<DocumentSnapshot>() {
                        @Override
public void onComplete(@NonNullTask<DocumentSnapshot>
task) {
                            if (task.isSuccessful()) {
                                default_account da =
task.getResult().toObject(default_account.class);
                                ((account_client)
(getApplicationContext())).setdefault_account(da);
                            }
                        }
                    });
                    Intent homepage = new Intent(Login.this,
MainUser.class);

homepage.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
                    startActivity(homepage);
                    finish();
                } else {
                    Toast.makeText(Login.this, "Anda belum
pernah login atau sesi sudah berahir, silahkan login kembali",
Toast.LENGTH_SHORT).show();
                }
            }
        };
    }

    @Override
    protected void onStart() {
        super.onStart();

FirebaseAuth.getInstance().addAuthStateListener(authStateListener
);
    }

    @Override
    protected void onStop() {
        super.onStop();
        if (authStateListener != null) {
```

```
127.
    FirebaseAuth.getInstance().removeAuthStateListener(authStateListe
    ner);
128.          }
129.      }
130. }
```

Main*Activity*.java

```
1. package com.example.tugasakhir_tracking_android;
2.
3. import android.content.Intent;
4. import android.os.Bundle;
5. import android.view.View;
6. import android.widget.EditText;
7. import android.widget.TextView;
8. import android.widget.Toast;
9.
10. import androidx.annotation.NonNull;
11. import androidx.appcompat.app.AppCompatActivity;
12. import androidx.cardview.widget.CardView;
13.
14. import com.google.android.gms.tasks.OnCompleteListener;
15. import com.google.android.gms.tasks.Task;
16. import com.google.firebase.auth.FirebaseAuth;
17. import com.google.firebase.auth.FirebaseUser;
18. import com.google.firebase.firestore.DocumentReference;
19. import com.google.firebase.firestore.DocumentSnapshot;
20. import com.google.firebase.firestore.FirebaseFirestore;
21.
22. public class MainActivity extends AppCompatActivity {
23.
24.     private CardView lokasi, tugas;
25.     private TextView nama, jabatan;
26.     private FirebaseFirestore firestore;
27.     private FirebaseAuth firebaseAuth;
28.     private FirebaseAuth.AuthStateListener stateListener;
29.     private DocumentReference documentReference;
30.
31.     @Override
32.     protected void onCreate(Bundle savedInstanceState) {
33.         super.onCreate(savedInstanceState);
34.         setContentView(R.layout.activity_main);
35.         nama = findViewById(R.id.tv_nama);
36.         jabatan = findViewById(R.id.tv_bagian);
37.
38.         lokasi = (CardView) findViewById(R.id.card_posisi);
39.         tugas = (CardView) findViewById(R.id.card_tugas);
40.
41.         setup_user();
42.
43.         lokasi.setOnClickListener(new View.OnClickListener() {
44.             @Override
45.             public void onClick(View v) {
46. Intent map = new Intent(MainActivity.this,
    MainPantau.class);
47. map.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK |
    Intent.FLAG_ACTIVITY_NEW_TASK);
```

```
48.                 startActivity(map);
49.                 finish();
50.             }
51.         });
52.         tugas.setOnClickListener(new View.OnClickListener() {
53.             @Override
54.             public void onClick(View v) {
55. Intent buat_tugas = new Intent(MainActivity.this,
    MainCreateTugas.class);
56.                 buat_tugas.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK
    | Intent.FLAG_ACTIVITY_CLEAR_TASK);
57.                 startActivity(buat_tugas);
58.                 finish();
59.             }
60.         });
61.     }
62.     private void setup_user() {
63.         firebaseAuth = FirebaseAuth.getInstance();
64.         stateListener = new FirebaseAuth.AuthStateListener() {
65.             @Override
66. public void onAuthStateChanged(@NonNull FirebaseAuth
    firebaseAuth) {
67.
68. FirebaseUser user =
    firebaseAuth.getCurrentUser();
69.
70.                 if (user != null) {
71.                     firestore = FirebaseFirestore.getInstance();
72. documentReference =
    firestore.collection("data_user").document(user.getUid());
73.
    documentReference.get().addOnCompleteListener(new
    OnCompleteListener<DocumentSnapshot>() {
74.                         @Override
75. public void onComplete(@NonNullTask<DocumentSnapshot>
    task) {
76.                             try {
77.                                 String karyawan =
    task.getResult().getString("nama");
78.                                 String bagian =
    task.getResult().getString("bagian");
79.
80.                                 nama.setText(karyawan);
81.                                 jabatan.setText(bagian);
82.                             } catch (Exception e) {
83.
    Toast.makeText(getApplicationContext(), "" + e,
    Toast.LENGTH_SHORT).show();
84.                             }
85.                         }
86.                     });
87.                 }
```

```
88.             }
89.         };
90.     }
91.
92.     @Override
93.     protected void onStart() {
94.         super.onStart();
95.         firebaseAuth.addAuthStateListener(stateListener);
96.     }
97.
98.     @Override
99.     protected void onStop() {
100.         super.onStop();
101.         if (stateListener != null){
102.
    firebaseAuth.removeAuthStateListener(stateListener);
103.         }
104.     }
105.
106.     @Override
107.     public void onBackPressed() {
108.         //super.onBackPressed();
109.         Intent back = new Intent(getApplicationContext(),
    MainUser.class);
110.         //back.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
111.         startActivity(back);
112.         finish();
113.     }
114. }
```

MainCreateTugas.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import androidx.annotation.NonNull;
4. import androidx.appcompat.app.AlertDialog;
5. import androidx.appcompat.app.AppCompatActivity;
6.
7. import android.app.DatePickerDialog;
8. import android.content.DialogInterface;
9. import android.content.Intent;
10. import android.os.Build;
11. import android.os.Bundle;
12. import android.view.View;
13. import android.widget.Button;
14. import android.widget.DatePicker;
15. import android.widget.EditText;
16. import android.widget.RadioButton;
17. import android.widget.Toast;
18.
19. import com.google.android.gms.tasks.OnCompleteListener;
20. import com.google.android.gms.tasks.OnSuccessListener;
21. import com.google.android.gms.tasks.Task;
22. import com.google.firebase.auth.FirebaseAuth;
23. import com.google.firebase.auth.FirebaseUser;
24. import com.google.firebase.firestore.CollectionReference;
25. import com.google.firebase.firestore.DocumentReference;
26. import com.google.firebase.firestore.DocumentSnapshot;
27. import com.google.firebase.firestore.FirebaseFirestore;
28. import com.google.firebase.firestore.Query;
29. import com.google.firebase.firestore.QueryDocumentSnapshot;
30. import com.google.firebase.firestore.QuerySnapshot;
31. import com.google.firebase.firestore.model.DocumentCollections;
32.
33. import java.text.SimpleDateFormat;
34. import java.util.Arrays;
35. import java.util.Calendar;
36. import java.util.HashMap;
37. import java.util.Locale;
38. import java.util.Map;
39.
40. public class MainCreateTugas extends AppCompatActivity {
41.     private EditText nama, alamat, tugas, pt, date;
42.     private Button input;
43.     private FirebaseFirestore firestore;
44.     private FirebaseAuth firebaseAuth;
45.     private FirebaseAuth.AuthStateListener stateListener;
46.     private DocumentReference documentReference;
47.     private CollectionReference collectionReference;
48.     private DatePickerDialog pickerDialog;
49.     private SimpleDateFormat format;
```

```
50.
51.     AlertDialog alertDialog;
52.
53.     @Override
54.     protected void onCreate(Bundle savedInstanceState) {
55.         super.onCreate(savedInstanceState);
56.         setContentView(R.layout.activity_main_create_tugas);
57.         firestore = FirebaseFirestore.getInstance();
58.         setup_user();
59.         nama = findViewById(R.id.et_nama);
60.         alamat = findViewById(R.id.et_alamat);
61.         tugas = findViewById(R.id.et_tugas);
62.         input = findViewById(R.id.btn_input_tugas);
63.         pt = findViewById(R.id.et_perusahaan);
64.         date = findViewById(R.id.et_date);
65.         format = new SimpleDateFormat("dd-MM-yyyy", Locale.US);
66.         date.setOnClickListener(new View.OnClickListener() {
67.             @Override
68.             public void onClick(View v) {
69.                 pick_date();
70.             }
71.         });
72.
73.         tugas.setOnClickListener(new View.OnClickListener() {
74.             @Override
75.             public void onClick(View v) {
76.                 daftar_karyawan();
77.             }
78.         });
79.
80.         input.setOnClickListener(new View.OnClickListener() {
81.             @Override
82.             public void onClick(View v) {
83.                 buat_tugas();
84.             }
85.         });
86.     }
87.
88.     private void pick_date() {
89.
90.         Calendar calendar = Calendar.getInstance();
91.         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
92. pickerDialog = new DatePickerDialog(this, new
    DatePickerDialog.OnDateSetListener() {
93.                 @Override
94. public void onDateSet(DatePicker view, int year,int month, int
    dayOfMonth) {
95.                     Calendar tgl = Calendar.getInstance();
96.                     tgl.set(year, month, dayOfMonth);
97.
98.                     date.setText(format.format(tgl.getTime()));
99.                 }
```

```
            }, calendar.get(Calendar.YEAR),
   calendar.get(Calendar.MONTH),
   calendar.get(Calendar.DAY_OF_MONTH));

            pickerDialog.show();
        }
    }

    private void buat_tugas() {
        String nama_karyawan, jenis_tugas, alamat_tugas,
   nama_pt, tgl_tugas;
        nama_karyawan = String.valueOf(nama.getText());
        jenis_tugas = String.valueOf(tugas.getText());
        alamat_tugas = String.valueOf(alamat.getText());
        nama_pt = String.valueOf(pt.getText());
        tgl_tugas = String.valueOf(date.getText());

        Map<String, Object> objectMap = new HashMap<>();
        objectMap.put("nama_karyawan", nama_karyawan);
        objectMap.put("jenis_tugas", jenis_tugas);
        objectMap.put("nama_perusahaan", nama_pt);
        objectMap.put("alamat_tujuan", alamat_tugas);
        objectMap.put("tgl", tgl_tugas);

        collectionReference = firestore.collection("id_task");

   collectionReference.add(objectMap).addOnCompleteListener(new
   OnCompleteListener<DocumentReference>() {
            @Override
            public void onComplete(@NonNull
   Task<DocumentReference> task) {
                if (task.isSuccessful()){
                    Toast.makeText(getApplicationContext(),
   "Tugas berhasil dimasukkan", Toast.LENGTH_SHORT).show();

                }
            }
        });

        AlertDialog.Builder pop_up = new
   AlertDialog.Builder(this);
        pop_up.setIcon(R.drawable.business_center)
                .setTitle("Tugas selesai dibuat")
                .setMessage("Tambah lagi tugas ?")
                .setPositiveButton("Ya", new
   DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog,
   int which) {
                        nama.setText("");
                        tugas.setText("");
                        alamat.setText("");
```

```
                        pt.setText("");
                        date.setText("");
                        dialog.cancel();
                    }
                })
                .setNegativeButton("Tidak", new
   DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog,
   int which) {
                        Intent intent = new
   Intent(MainCreateTugas.this, MainActivity.class);

   intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        startActivity(intent);
                        finish();
                    }
                });
        AlertDialog dialog = pop_up.create();
        dialog.show();
    }

    private void daftar_karyawan() {
        String[] daftar_tugas = new String[]{"Visit", "Kirim
   Barang"};
        AlertDialog.Builder message = new
   AlertDialog.Builder(this);
        message.setTitle("Pilih Penugasan :")
                .setSingleChoiceItems(daftar_tugas, -1, new
   DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog,
   int which) {
                        switch (which) {
                            case 0:
                                tugas.setText("Visit");
                                break;
                            case 1:
                                tugas.setText("Kirim Barang");
                                break;
                        }
                        alertDialog.dismiss();
                    }
                })
                .setIcon(R.drawable.business_center);
        alertDialog = message.create();
        alertDialog.show();
    }


    private void setup_user() {
        firebaseAuth = FirebaseAuth.getInstance();
```

```
186.          stateListener = new FirebaseAuth.AuthStateListener() {
187.              @Override
188.              public void onAuthStateChanged(@NonNull
    FirebaseAuth firebaseAuth) {
189.
190.                  FirebaseUser user =
    firebaseAuth.getCurrentUser();
191.
192.                  if (user != null) {
193.                      firestore =
    FirebaseFirestore.getInstance();
194.                      documentReference =
    firestore.collection("data_user").document(user.getUid());
195.
196.                  }
197.              }
198.          };
199.      }
200.
201.      @Override
202.      protected void onStart() {
203.          super.onStart();
204.          firebaseAuth.addAuthStateListener(stateListener);
205.      }
206.
207.      @Override
208.      protected void onStop() {
209.          super.onStop();
210.          if (stateListener != null) {
211.
    firebaseAuth.removeAuthStateListener(stateListener);
212.          }
213.      }
214.
215.      @Override
216.      public void onBackPressed() {
217.          //super.onBackPressed();
218.          Intent back = new Intent(getApplicationContext(),
    MainActivity.class);
219.          //back.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
220.          startActivity(back);
221.          finish();
222.      }
223. }
224.
```

MainPantau.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import androidx.annotation.NonNull;
4. import androidx.annotation.Nullable;
5. import androidx.appcompat.app.AlertDialog;
6. import androidx.appcompat.app.AppCompatActivity;
7.
8. import android.app.DatePickerDialog;
9. import android.content.Intent;
10. import android.location.Address;
11. import android.location.Geocoder;
12. import android.os.Build;
13. import android.os.Bundle;
14. import android.util.Log;
15. import android.view.View;
16. import android.widget.Button;
17. import android.widget.DatePicker;
18. import android.widget.EditText;
19. import android.widget.SearchView;
20. import android.widget.Toast;
21.
22. import com.google.android.gms.maps.CameraUpdateFactory;
23. import com.google.android.gms.maps.GoogleMap;
24. import com.google.android.gms.maps.OnMapReadyCallback;
25. import com.google.android.gms.maps.SupportMapFragment;
26. import com.google.android.gms.maps.model.LatLng;
27. import com.google.android.gms.maps.model.Marker;
28. import com.google.android.gms.maps.model.MarkerOptions;
29. import com.google.android.gms.tasks.OnCompleteListener;
30. import com.google.android.gms.tasks.OnSuccessListener;
31. import com.google.android.gms.tasks.Task;
32. import com.google.firebase.auth.FirebaseAuth;
33. import com.google.firebase.auth.FirebaseUser;
34. import com.google.firebase.firestore.CollectionReference;
35. import com.google.firebase.firestore.DocumentReference;
36. import com.google.firebase.firestore.DocumentSnapshot;
37. import com.google.firebase.firestore.EventListener;
38. import com.google.firebase.firestore.FirebaseFirestore;
39. import com.google.firebase.firestore.FirebaseFirestoreException;
40. import com.google.firebase.firestore.GeoPoint;
41. import com.google.firebase.firestore.Query;
42. import com.google.firebase.firestore.QueryDocumentSnapshot;
43. import com.google.firebase.firestore.QuerySnapshot;
44.
45. import java.text.SimpleDateFormat;
46. import java.util.ArrayList;
47. import java.util.Calendar;
48. import java.util.HashMap;
49. import java.util.List;
```

```
50. import java.util.Locale;
51. import java.util.Map;
52.
53. public class MainPantau extends AppCompatActivity {
54.     private SupportMapFragment supportMapFragment;
55.     private FirebaseFirestore firestore;
56.     private DocumentReference reference;
57.     private FirebaseAuth firebaseAuth;
58.     private FirebaseAuth.AuthStateListener stateListener;
59.     private FirebaseUser firebaseUser;
60.     private CollectionReference collectionReference;
61.     private EditText nama, tgl_update;
62.     private Button cari;
63.     private DatePickerDialog pickerDialog;
64.     private SimpleDateFormat format;
65.
66.     @Override
67.     protected void onCreate(Bundle savedInstanceState) {
68.         super.onCreate(savedInstanceState);
69.         setContentView(R.layout.activity_main_pantau);
70. supportMapFragment = (SupportMapFragment)
    getSupportFragmentManager().findFragmentById(R.id.map_all);
71.
72.         nama = findViewById(R.id.et_cari_nama);
73.         tgl_update = findViewById(R.id.et_cari_tgl);
74.         cari = findViewById(R.id.btn_cari);
75.         setup_user();
76.
77.         tgl_update.setOnClickListener(new View.OnClickListener()
    {
78.             @Override
79.             public void onClick(View v) {
80.                 pick_date();
81.             }
82.         });
83.
84.         firestore = FirebaseFirestore.getInstance();
85. collectionReference =
    firestore.collection("data_location");
86.         firebaseUser = firebaseAuth.getCurrentUser();
87.         cari.setOnClickListener(new View.OnClickListener() {
88.             @Override
89.             public void onClick(View v) {
90. supportMapFragment.getMapAsync(new
    OnMapReadyCallback() {
91.                     @Override
92. public void onMapReady(@NonNull GoogleMapgoogleMap) {
93.                         googleMap.clear();
94.                         String nama_karyawan, tgl_map;
95.                         nama_karyawan =
    String.valueOf(nama.getText());
```

```
                        tgl_map =
    String.valueOf(tgl_update.getText());
                        //reference =
    firestore.collection("ColRef_location").document(firebaseUser.get
    Uid());

collectionReference.whereEqualTo("nama",
    nama_karyawan).get().addOnCompleteListener(new
    OnCompleteListener<QuerySnapshot>() {
                            @Override
                            public void onComplete(@NonNull
    Task<QuerySnapshot> task) {
                                try {
                                    if (task.isSuccessful()) {
                                        for
    (QueryDocumentSnapshot documentSnapshot : task.getResult()) {

    //Toast.makeText(getApplicationContext(), "Berhasil",
    Toast.LENGTH_SHORT).show();
                                            List<Object> point
    = (List<Object>) documentSnapshot.get(tgl_map);
                                            int length =
    point.size();
                                            if (length == 0) {
                                                return;
                                            }

                                            for (int i = 0; i <
    length; i++) {
                                                GeoPoint lokasi
    = (GeoPoint) point.get(i);
                                                double latitude
    = lokasi.getLatitude();
                                                double
    longitude = lokasi.getLongitude();
                                                LatLng latLng =
    new LatLng(latitude, longitude);
                                                Geocoder
     geocoder = new Geocoder(MainPantau.this, Locale.getDefault());
                                                List<Address>
    addresses = geocoder.getFromLocation(latitude, longitude, 1);

                                                String tempat =
    addresses.get(0).getAddressLine(0);

    googleMap.addMarker(new
    MarkerOptions().position(latLng).title("Lokasi" + (i +
    1)).snippet(tempat));

    googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng,
    15));

```

```
124.                                                }
125.
    googleMap.setOnInfoWindowClickListener(new
    GoogleMap.OnInfoWindowClickListener() {
126.                                                    @Override
127.                                                    public void
    onInfoWindowClick(@NonNull Marker marker) {
128.
    AlertDialog.Builder builder = new
    AlertDialog.Builder(MainPantau.this);
129.
    builder.setTitle("Info Lokasi")
130.                                                                .se
    tMessage(marker.getSnippet());
131.                                                        AlertDialog
    alertDialog = builder.create();
132.
    alertDialog.show();
133.                                                    }
134.                                                });
135.
136.                                            }
137.                                        }
138.                                    } catch (Exception exception) {
139.
    exception.printStackTrace();
140.
    Toast.makeText(getApplicationContext(), "" + exception,
    Toast.LENGTH_LONG).show();
141.                                    }
142.                                }
143.                            });
144.                        }
145.                    });
146.                }
147.            });
148.
149.
150.      }
151.
152.      private void setup_user() {
153.          firebaseAuth = FirebaseAuth.getInstance();
154.          stateListener = new FirebaseAuth.AuthStateListener() {
155.              @Override
156.              public void onAuthStateChanged(@NonNull
    FirebaseAuth firebaseAuth) {
157.
158.                  FirebaseUser user =
    firebaseAuth.getCurrentUser();
159.
160.                  if (user != null) {
```

```
                    firestore = FirebaseFirestore.getInstance();
                    reference = firestore.collection("data_user").document(user.getUid());

                }
            }
        };
    }

    private void pick_date() {
        format = new SimpleDateFormat("dd-MM-yyyy", Locale.US);
        Calendar calendar = Calendar.getInstance();
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
            pickerDialog = new DatePickerDialog(this, new DatePickerDialog.OnDateSetListener() {
                @Override
                public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
                    Calendar tgl = Calendar.getInstance();
                    tgl.set(year, month, dayOfMonth);

                    tgl_update.setText(format.format(tgl.getTime()));
                }
            }, calendar.get(Calendar.YEAR), calendar.get(Calendar.MONTH), calendar.get(Calendar.DAY_OF_MONTH));

            pickerDialog.show();
        }
    }

    @Override
    protected void onStart() {
        super.onStart();
        firebaseAuth.addAuthStateListener(stateListener);
    }

    @Override
    protected void onStop() {
        super.onStop();
        if (stateListener != null) {
            firebaseAuth.removeAuthStateListener(stateListener);
        }
    }

    @Override
    public void onBackPressed() {
        //super.onBackPressed();
```

```
204.          Intent back = new Intent(getApplicationContext(),
   MainUser.class);
205.          //back.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
206.          startActivity(back);
207.          finish();
208.      }
209. }
```

MainUser.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.content.DialogInterface;
4. import android.content.Intent;
5. import android.os.Bundle;
6. import android.view.View;
7. import android.widget.TextView;
8. import android.widget.Toast;
9.
10. import androidx.annotation.NonNull;
11. import androidx.appcompat.app.AlertDialog;
12. import androidx.appcompat.app.AppCompatActivity;
13. import androidx.fragment.app.FragmentPagerAdapter;
14. import androidx.viewpager.widget.ViewPager;
15.
16. import com.google.android.gms.tasks.OnCompleteListener;
17. import com.google.android.gms.tasks.OnSuccessListener;
18. import com.google.android.gms.tasks.Task;
19. import
    com.google.android.material.floatingactionbutton.FloatingActionBu
    tton;
20. import com.google.android.material.tabs.TabLayout;
21. import com.google.firebase.auth.FirebaseAuth;
22. import com.google.firebase.auth.FirebaseUser;
23. import com.google.firebase.firestore.DocumentReference;
24. import com.google.firebase.firestore.DocumentSnapshot;
25. import com.google.firebase.firestore.FirebaseFirestore;
26.
27. public class MainUser extends AppCompatActivity {
28.     private TabLayout tabLayout;
29.     private ViewPager viewPager;
30.     private TextView jabatan, nama;
31.     private default_account defaultAccount;
32.     private FirebaseFirestore firestore;
33.     private FirebaseAuth firebaseAuth;
34.     private FirebaseAuth.AuthStateListener stateListener;
35.     private DocumentReference documentReference;
36.     private FloatingActionButton floating_button;
37.
38.
39.     @Override
40.     protected void onCreate(Bundle savedInstanceState) {
41.         super.onCreate(savedInstanceState);
42.         setContentView(R.layout.activity_main_user);
43.         tabLayout = (TabLayout) findViewById(R.id.tl_menu);
44.         viewPager = (ViewPager) findViewById(R.id.vp);
45.         jabatan = (TextView) findViewById(R.id.tv_jabatan);
46.         nama = (TextView) findViewById(R.id.tv_nama_karyawan);
```

```
47. floating_button = (FloatingActionButton)
    findViewById(R.id.float_bt);
48.         setup_user();
49.
50.         tabLayout.setupWithViewPager(viewPager);
51.
52.         View_Page_Adapter view_page_adapter = new
    View_Page_Adapter(getSupportFragmentManager(),
    FragmentPagerAdapter.BEHAVIOR_RESUME_ONLY_CURRENT_FRAGMENT);
53.
54. view_page_adapter.addFragment(new
    fragment_daftar_tugas(), "Informasi");
55. view_page_adapter.addFragment(new fragment_map_user(),"Lokasi
    Saya");
56.
57.         viewPager.setAdapter(view_page_adapter);
58.
59. floating_button.setOnClickListener(new
    View.OnClickListener() {
60.             @Override
61.             public void onClick(View v) {
62. Intent tool = new Intent(MainUser.this,
    MainActivity.class);
63. tool.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
    Intent.FLAG_ACTIVITY_CLEAR_TASK);
64.                 startActivity(tool);
65.                 finish();
66.             }
67.         });
68.
69.     }
70.
71.     private void setup_user() {
72.         firebaseAuth = FirebaseAuth.getInstance();
73.         stateListener = new FirebaseAuth.AuthStateListener() {
74.             @Override
75. public void onAuthStateChanged(@NonNull FirebaseAuth
    firebaseAuth) {
76.
77. FirebaseUser user =
    firebaseAuth.getCurrentUser();
78.
79.                 if (user != null) {
80.                     firestore = FirebaseFirestore.getInstance();
81. documentReference =
    firestore.collection("data_user").document(user.getUid());
82.
    documentReference.get().addOnCompleteListener(new
    OnCompleteListener<DocumentSnapshot>() {
83.                         @Override
84. public void onComplete(@NonNullTask<DocumentSnapshot>
    task) {
```

```
85.                             try {
86.                                 String karyawan =
   task.getResult().getString("nama");
87.                                 String bagian =
   task.getResult().getString("bagian");
88.
89.                                 if
   (task.getResult().getString("level").equals("staff")){
90.
   floating_button.setVisibility(View.INVISIBLE);
91.                                 } else {
92.
   floating_button.setVisibility(View.VISIBLE);
93.                                 }
94.
95.                                 nama.setText(karyawan);
96.                                 jabatan.setText(bagian);
97.                             } catch (Exception e) {
98.
   Toast.makeText(getApplicationContext(), "" + e,
   Toast.LENGTH_SHORT).show();
99.                             }
100.                        }
101.                    });
102.                }
103.            }
104.        };
105.    }
106.
107.    @Override
108.    protected void onStart() {
109.        super.onStart();
110.        firebaseAuth.addAuthStateListener(stateListener);
111.    }
112.
113.    @Override
114.    protected void onStop() {
115.        super.onStop();
116.        if (stateListener != null) {
117.
   firebaseAuth.removeAuthStateListener(stateListener);
118.        }
119.
120.    }
121.
122.    @Override
123.    public void onBackPressed() {
124.        //super.onBackPressed();
125.        AlertDialog.Builder pesan = new
   AlertDialog.Builder(MainUser.this);
126.        pesan.setTitle("Keluar Aplikasi")
127.                .setMessage("Yakin keluar dari aplikasi ?")
```

```
                .setIcon(R.drawable.ic_logo_sri_indah_labetama_
   __fie_hd_trans_coba)
                .setPositiveButton("Ya", new
   DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog,
   int which) {
                        MainUser.super.onBackPressed();
                    }
                })
                .setNegativeButton("Tidak", new
   DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog,
   int which) {
                        dialog.cancel();
                    }
                });
        AlertDialog pop_up = pesan.create();
        pop_up.show();
    }
}
```

SplashScreen.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import androidx.appcompat.app.AppCompatActivity;
4.
5. import android.content.Intent;
6. import android.os.Bundle;
7. import android.os.Handler;
8. import android.widget.ImageView;
9. import android.widget.TextView;
10.
11.public class SplashScreen extends AppCompatActivity {
12.
13.    private ImageView logo;
14.
15.    @Override
16.    protected void onCreate(Bundle savedInstanceState) {
17.        super.onCreate(savedInstanceState);
18.        setContentView(R.layout.activity_splash_screen);
19.        logo = (ImageView) findViewById(R.id.img_logo_apk);
20.        new Handler().postDelayed(new Runnable() {
21.            @Override
22.            public void run() {
23.                Intent login = new Intent(SplashScreen.this,
   Login.class);
24.                login.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
25.                startActivity(login);
26.                finish();
27.            }
28.        }, 5000);
29.    }
30.}
```

Tugas_adapter.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import android.Manifest;
4. import android.content.pm.PackageManager;
5. import android.view.LayoutInflater;
6. import android.view.View;
7. import android.view.ViewGroup;
8. import android.widget.TextView;
9.
10. import androidx.annotation.NonNull;
11. import androidx.core.app.ActivityCompat;
12. import androidx.recyclerview.widget.RecyclerView;
13.
14. import com.firebase.ui.firestore.FirestoreRecyclerAdapter;
15. import com.firebase.ui.firestore.FirestoreRecyclerOptions;
16. import
    com.google.android.gms.location.FusedLocationProviderClient;
17. import com.google.android.gms.location.LocationServices;
18. import com.google.firebase.firestore.DocumentSnapshot;
19. import com.google.firebase.firestore.FirebaseFirestore;
20. import com.google.firebase.firestore.GeoPoint;
21.
22. import java.text.SimpleDateFormat;
23. import java.util.Arrays;
24. import java.util.Date;
25. import java.util.HashMap;
26. import java.util.Map;
27.
28. public class tugas_adapter extends
    FirestoreRecyclerAdapter<file_tugas, tugas_adapter.tugas_holder>
    {
29.     private FirebaseFirestore firestore;
30. //private CollectionReference reference =
    firestore.collection("data_location");
31.     private FusedLocationProviderClient client;
32.     private OnItemClickListener listener;
33.
34.
35.     public tugas_adapter(@NonNull
    FirestoreRecyclerOptions<file_tugas> options) {
36.
37.         super(options);
38.     }
39.
40.     @Override
41. protected void onBindViewHolder(@NonNull tugas_holder holder,int
    position, @NonNull file_tugas model) {
42.         holder.judul_tugas.setText(model.getjenis_tugas());
43.         holder.pt.setText(model.getnama_perusahaan());
```

```
        holder.alamat.setText(model.getalamat_tujuan());

    }

    @NonNull
    @Override
public tugas_holder onCreateViewHolder(@NonNull ViewGroup
parent, int viewType) {
View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.layout_
recycle_view_tugas, parent, false);
        return new tugas_holder(view);
    }
/*
    @Override
    public void onDataChanged() {
        super.onDataChanged();
    }

 */


    public void konfirm_tugas(int position) {
        Map<String, Object> status = new HashMap<>();
        status.put("status_tugas", "selesai");
        getSnapshots().getSnapshot(position).getReference()
                .update(status);

        //submit_location();

    }

      class tugas_holder extends RecyclerView.ViewHolder {
        TextView judul_tugas;
        TextView alamat;
        TextView pt;
       // CardView container_tugas;

        public tugas_holder(@NonNull View itemView) {
            super(itemView);
judul_tugas =
itemView.findViewById(R.id.tv_judul_tugas);
            alamat = itemView.findViewById(R.id.tv_alamat);
            pt = itemView.findViewById(R.id.tv_nama_perusahaan);
//container_tugas =
itemView.findViewById(R.id.cv_tugas);
itemView.setOnClickListener(new
View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    int position = getAdapterPosition();
```

```
89.                     if (position != RecyclerView.NO_POSITION &&
   listener != null){
90.
   listener.OnItemClick(getSnapshots().getSnapshot(position),
   position);
91.
92.                         /*
93. AlertDialog.Builder prompt = new
   AlertDialog.Builder(v.getContext());
94. prompt.setTitle("Konfirmasi selesaitugas ?")
95.                                 .setMessage("Lokasi anda akan
   disimpan di database")
96.                                 .setPositiveButton("Ya", new
   DialogInterface.OnClickListener() {
97.                                     @Override
98.                                     public void
   onClick(DialogInterface dialog, int which) {
99.

   v.setVisibility(View.GONE);
100.                                     }
101.                                 })
102.                                 .setNegativeButton("Tidak", new
   DialogInterface.OnClickListener() {
103.                                     @Override
104.                                     public void
   onClick(DialogInterface dialog, int which) {
105.                                         dialog.cancel();
106.                                     }
107.                                 });
108.                         AlertDialog dialog = prompt.create();
109.                         dialog.show();
110.
111.                          */
112.                     }
113.                 }
114.             });
115.         }
116.     }
117.     public interface OnItemClickListener{
118.         void OnItemClick(DocumentSnapshot documentSnapshot, int
   position);
119.     }
120.     public void setOnClickListener(OnItemClickListener
   listener){
121.
122.         this.listener = listener;
123.
124.     }
125. }
```

View_Page_Adapter.java

```
1. package com.example.tugasakhirtrackingandroid;
2.
3. import androidx.annotation.NonNull;
4. import androidx.annotation.Nullable;
5. import androidx.fragment.app.Fragment;
6. import androidx.fragment.app.FragmentManager;
7. import androidx.fragment.app.FragmentPagerAdapter;
8.
9. import java.util.ArrayList;
10.
11. public class View_Page_Adapter extends FragmentPagerAdapter {
12.
13. private final ArrayList<Fragment> fragmentArrayList = new
    ArrayList<>();
14. private final ArrayList<String> stringArrayList = new
    ArrayList<>();
15.
16. public View_Page_Adapter(@NonNull FragmentManager fm, int
    behavior) {
17.
18.         super(fm, behavior);
19.     }
20.
21.     @NonNull
22.     @Override
23.     public Fragment getItem(int position) {
24.
25.         return fragmentArrayList.get(position);
26.     }
27.
28.     @Override
29.     public int getCount() {
30.
31.         return fragmentArrayList.size();
32.     }
33.
34.     public void addFragment(Fragment fragment, String s){
35.         fragmentArrayList.add(fragment);
36.         stringArrayList.add(s);
37.     }
38.
39.     @Nullable
40.     @Override
41.     public CharSequence getPageTitle(int position) {
42.         return stringArrayList.get(position);
43.     }
44. }
```

Lampiran 2

*activity*_login.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#29b6f6"
8.     tools:context=".Login">
9.
10.    <LinearLayout
11.        android:id="@+id/linearLayout3"
12.        android:layout_width="match_parent"
13.        android:layout_height="0dp"
14.        android:orientation="horizontal"
15.        android:paddingTop="18dp"
16.        android:paddingBottom="18dp"
17.        app:layout_constraintEnd_toEndOf="parent"
18.        app:layout_constraintStart_toStartOf="parent"
19.        app:layout_constraintTop_toTopOf="parent">
20.
21.        <TextView
22.            android:id="@+id/textView15"
23.            android:layout_width="wrap_content"
24.            android:layout_height="70dp"
25.
   android:layout_gravity="center_horizontal|center_vertical"
26.            android:layout_marginStart="40dp"
27.            android:fontFamily="@font/didact_gothic"
28.            android:gravity="center"
29.            android:text="SIL Track v1.0"
30.            android:textColor="#000000"
31.            android:textSize="34sp" />
32.
33.        <ImageView
34.            android:id="@+id/imageView7"
35.            android:layout_width="70dp"
36.            android:layout_height="70dp"
37.
   android:layout_gravity="center_horizontal|center_vertical"
38.            android:layout_marginLeft="40dp"
39.            app:srcCompat="@drawable/logo_tracking" />
40.    </LinearLayout>
41.
42.    <LinearLayout
43.        android:id="@+id/linearLayout"
44.        android:layout_width="match_parent"
```

```
45.         android:layout_height="0dp"
46.         android:layout_marginTop="16dp"
47.         android:background="@drawable/border_radius_login"
48.         android:elevation="8dp"
49.         android:orientation="vertical"
50.
51.         android:paddingLeft="40dp"
52.         android:paddingTop="20dp"
53.         android:paddingRight="40dp"
54.         android:paddingBottom="20dp"
55.         app:layout_constraintBottom_toBottomOf="parent"
56.         app:layout_constraintEnd_toEndOf="parent"
57.         app:layout_constraintHorizontal_bias="0.454"
58.         app:layout_constraintStart_toStartOf="parent"
59.         app:layout_constraintTop_toBottomOf="@+id/linearLayout3">
60.
61.         <EditText
62.             android:id="@+id/et_email"
63.             android:layout_width="match_parent"
64.             android:layout_height="wrap_content"
65.             android:backgroundTint="#5C5959"
66.             android:ems="10"
67.             android:fontFamily="@font/didact_gothic"
68.             android:hint="Email"
69.             android:inputType="textPersonName"
70.             android:textColor="#171717"
71.             android:textColorHighlight="#5E5A5A"
72.             android:textColorHint="#706969"
73.             android:textSize="20sp" />
74.
75.         <EditText
76.             android:id="@+id/et_passwd"
77.             android:layout_width="match_parent"
78.             android:layout_height="wrap_content"
79.             android:backgroundTint="#5C5959"
80.             android:ems="10"
81.             android:fontFamily="@font/didact_gothic"
82.             android:hint="Password"
83.             android:inputType="textPassword"
84.             android:paddingTop="10dp"
85.             android:textColor="#171717"
86.             android:textColorHint="#706969"
87.             android:textSize="20sp" />
88.
89.         <Button
90.             android:id="@+id/btn_login"
91.             android:layout_width="150dp"
92.             android:layout_height="wrap_content"
93.             android:layout_gravity="center_horizontal"
94.             android:layout_marginTop="30dp"
95.             android:background="@drawable/border_radius_all"
96.             android:fontFamily="@font/didact_gothic"
```

```
97.             android:text="login"
98.             android:textSize="18sp" />
99.
100.         <ProgressBar
101.             android:id="@+id/progbar_loading"
102.             style="?android:attr/progressBarStyle"
103.             android:layout_width="match_parent"
104.             android:layout_height="wrap_content"
105.             android:layout_marginStart="40dp"
106.             android:layout_marginLeft="40dp"
107.             android:layout_marginTop="24dp"
108.             android:layout_marginEnd="40dp"
109.             android:layout_marginRight="40dp"
110.             app:layout_constraintEnd_toEndOf="parent"
111.             app:layout_constraintStart_toStartOf="parent"
112.
   app:layout_constraintTop_toBottomOf="@+id/btn_login" />
113.
114.     </LinearLayout>
115.
116. </androidx.constraintlayout.widget.ConstraintLayout>
117.
```

***activity*_main.xml**

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#FFFFFF"
8.     tools:context=".Main*Activity*">
9.
10.    <androidx.constraintlayout.widget.ConstraintLayout
11.        android:id="@+id/header"
12.        android:layout_width="match_parent"
13.        android:layout_height="wrap_content"
14.        android:background="#FF8F00"
15.        android:elevation="10dp"
16.        app:layout_constraintEnd_toEndOf="parent"
17.        app:layout_constraintStart_toStartOf="parent"
18.        app:layout_constraintTop_toTopOf="parent">
19.
20.        <androidx.constraintlayout.widget.ConstraintLayout
21.            android:id="@+id/constraintLayout2"
22.            android:layout_width="wrap_content"
23.            android:layout_height="wrap_content"
24.            android:background="@drawable/bg_ellipse"
25.            android:padding="20dp"
26.            app:layout_constraintBottom_toBottomOf="parent"
27.            app:layout_constraintStart_toStartOf="parent"
28.            app:layout_constraintTop_toTopOf="parent">
29.
30.            <TextView
31.                android:id="@+id/textView7"
32.                android:layout_width="wrap_content"
33.                android:layout_height="wrap_content"
34.                android:text="Halaman Penugasan"
35.                android:textColor="@color/black"
36.                android:textSize="24sp"
37.                app:layout_constraintStart_toStartOf="parent"
38.                app:layout_constraintTop_toTopOf="parent" />
39.
40.            <TextView
41.                android:id="@+id/textView8"
42.                android:layout_width="wrap_content"
43.                android:layout_height="wrap_content"
44.                android:layout_marginTop="8dp"
45. android:text="Silahkan pilih menu penugasan yangada"
46.                android:textColor="#000000"
47.                app:layout_constraintStart_toStartOf="parent"
```

```
48.
   app:layout_constraintTop_toBottomOf="@+id/textView7" />
49.         </androidx.constraintlayout.widget.ConstraintLayout>
50.
51.         <ImageView
52.             android:id="@+id/imageView4"
53.             android:layout_width="0dp"
54.             android:layout_height="0dp"
55.             app:layout_constraintBottom_toBottomOf="parent"
56.             app:layout_constraintEnd_toEndOf="parent"
57.
   app:layout_constraintStart_toEndOf="@+id/constraintLayout2"
58.             app:layout_constraintTop_toTopOf="parent"
59.             app:srcCompat="@drawable/business_center"
60.             tools:ignore="ImageContrastCheck" />
61.
62.     </androidx.constraintlayout.widget.ConstraintLayout>
63.
64.     <LinearLayout
65.         android:id="@+id/linearLayout4"
66.         android:layout_width="match_parent"
67.         android:layout_height="300dp"
68.         android:isScrollContainer="true"
69.         android:orientation="horizontal"
70.         android:paddingRight="20dp"
71.         app:layout_constraintEnd_toEndOf="parent"
72.         app:layout_constraintStart_toStartOf="parent"
73.         app:layout_constraintTop_toBottomOf="@+id/header">
74.
75.         <androidx.cardview.widget.CardView
76.             android:layout_width="200dp"
77.             android:layout_height="match_parent"
78.             android:layout_marginStart="20dp"
79.             android:layout_marginTop="20dp"
80.             android:layout_marginBottom="20dp"
81.             android:padding="20dp"
82.             app:cardBackgroundColor="#00b0ff"
83.             app:cardCornerRadius="8dp"
84.             app:cardElevation="5dp">
85.
86.             <LinearLayout
87.                 android:layout_width="match_parent"
88.                 android:layout_height="match_parent"
89.                 android:gravity="center_vertical"
90.                 android:orientation="vertical"
91.                 android:padding="10dp">
92.
93.                 <TextView
94.                     android:id="@+id/textView13"
95.                     android:layout_width="match_parent"
96.                     android:layout_height="wrap_content"
97.                     android:gravity="center_horizontal"
```

```
                    android:text="Informasi Akun"
                    android:textColor="#121010"
                      android:textSize="20sp" />

                <ImageView
                    android:id="@+id/imageView5"
                    android:layout_width="100dp"
                    android:layout_height="100dp"
   android:layout_gravity="center_horizontal|center_vertical"
                    android:layout_margin="10dp"
                    app:srcCompat="@drawable/receptionist" />

                <TextView
                    android:id="@+id/tv_nama"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:gravity="center_horizontal"
                    android:padding="5dp"
                    android:text="nama karyawan"
                    android:textColor="#121010"
                    android:textSize="16sp" />

                <TextView
                    android:id="@+id/tv_bagian"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:gravity="center_horizontal"
                    android:padding="5dp"
                    android:text="bagian"
                    android:textColor="#121010"
                    android:textSize="16sp" />
            </LinearLayout>
        </androidx.cardview.widget.CardView>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginTop="20dp"
            android:layout_marginBottom="20dp"
            android:orientation="vertical"
            android:paddingLeft="10dp">

            <androidx.cardview.widget.CardView
                android:id="@+id/card_posisi"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:clickable="true"
                android:padding="20dp"
                app:cardBackgroundColor="#FF8F00"
                app:cardCornerRadius="8dp"
                app:cardElevation="5dp">
```

```
149.
150.                 <LinearLayout
151.                     android:layout_width="match_parent"
152.                     android:layout_height="match_parent"
153.                     android:orientation="vertical"
154.                     android:padding="10dp">
155.
156.                     <ImageView
157.                         android:id="@+id/imageView6"
158.                         android:layout_width="70dp"
159.                         android:layout_height="70dp"
160.
   android:layout_gravity="center_horizontal"
161.                         app:srcCompat="@drawable/logo_lokasi"
   />
162.
163.                     <TextView
164.                         android:id="@+id/textView17"
165.                         android:layout_width="match_parent"
166.                         android:layout_height="wrap_content"
167.                         android:gravity="center_horizontal"
168.                         android:text="Posisi \nKaryawan"
169.                         android:textColor="#FFFFFF"
170.                         android:textSize="16sp" />
171.                 </LinearLayout>
172.             </androidx.cardview.widget.CardView>
173.
174.             <androidx.cardview.widget.CardView
175.                 android:id="@+id/card_tugas"
176.                 android:layout_width="match_parent"
177.                 android:layout_height="wrap_content"
178.                 android:layout_marginTop="10dp"
179.                 android:clickable="true"
180.                 android:padding="20dp"
181.                 app:cardBackgroundColor="#FF8F00"
182.                 app:cardCornerRadius="8dp"
183.                 app:cardElevation="5dp">
184.
185.                 <LinearLayout
186.                     android:layout_width="match_parent"
187.                     android:layout_height="match_parent"
188.                     android:orientation="vertical"
189.                     android:padding="5dp">
190.
191.                     <ImageView
192.                         android:id="@+id/imageView8"
193.                         android:layout_width="70dp"
194.                         android:layout_height="70dp"
195.
   android:layout_gravity="center_horizontal"
196.
   app:srcCompat="@drawable/ic_baseline_add" />
```

```
197.
198.                     <TextView
199.                         android:id="@+id/textView18"
200.                         android:layout_width="match_parent"
201.                         android:layout_height="wrap_content"
202.                         android:gravity="center_horizontal"
203.                         android:text="Tambah \nTugas"
204.                         android:textColor="#ffffff"
205.                         android:textSize="16sp" />
206.                 </LinearLayout>
207.             </androidx.cardview.widget.CardView>
208.
209.         </LinearLayout>
210.     </LinearLayout>
211.
212.     <ImageView
213.         android:id="@+id/imageView9"
214.         android:layout_width="150dp"
215.         android:layout_height="150dp"
216.         android:layout_marginTop="16dp"
217.         app:layout_constraintEnd_toEndOf="parent"
218.         app:layout_constraintStart_toStartOf="parent"
219.
    app:layout_constraintTop_toBottomOf="@+id/linearLayout4"
220.
    app:srcCompat="@drawable/ic_logo_sri_indah_labetama_fie_hd_tran
    s_coba" />
221. </androidx.constraintlayout.widget.ConstraintLayout>
222.
```

***activity*_main_create_tugas.xml**

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#ffffff"
8.     tools:context=".MainCreateTugas">
9.
10.    <androidx.constraintlayout.widget.ConstraintLayout
11.        android:id="@+id/header"
12.        android:layout_width="match_parent"
13.        android:layout_height="wrap_content"
14.        android:background="#FF8F00"
15.        android:elevation="8dp"
16.        app:layout_constraintEnd_toEndOf="parent"
17.        app:layout_constraintStart_toStartOf="parent"
18.        app:layout_constraintTop_toTopOf="parent">
19.
20.        <androidx.constraintlayout.widget.ConstraintLayout
21.            android:id="@+id/constraintLayout2"
22.            android:layout_width="wrap_content"
23.            android:layout_height="wrap_content"
24.            android:background="@drawable/bg_ellipse"
25.            android:padding="20dp"
26.            app:layout_constraintBottom_toBottomOf="parent"
27.            app:layout_constraintStart_toStartOf="parent"
28.            app:layout_constraintTop_toTopOf="parent">
29.
30.            <TextView
31.                android:id="@+id/textView7"
32.                android:layout_width="wrap_content"
33.                android:layout_height="wrap_content"
34.                android:fontFamily="@font/didact_gothic"
35.                android:text="Tambah Tugas"
36.                android:textColor="@color/black"
37.                android:textSize="24sp"
38.                app:layout_constraintStart_toStartOf="parent"
39.                app:layout_constraintTop_toTopOf="parent" />
40.
41.            <TextView
42.                android:id="@+id/textView8"
43.                android:layout_width="wrap_content"
44.                android:layout_height="wrap_content"
45.                android:layout_marginTop="8dp"
46.                android:fontFamily="@font/didact_gothic"
47. android:text="Form di bawah untuk menambahkantugas baru"
```

```
                android:textColor="#000000"
                app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintTop_toBottomOf="@+id/textView7" />
        </androidx.constraintlayout.widget.ConstraintLayout>

        <ImageView
            android:id="@+id/imageView4"
            android:layout_width="0dp"
            android:layout_height="0dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"

    app:layout_constraintStart_toEndOf="@+id/constraintLayout2"
            app:layout_constraintTop_toTopOf="parent"
            app:srcCompat="@drawable/ic_baseline_add"
            tools:ignore="ImageContrastCheck" />

    </androidx.constraintlayout.widget.ConstraintLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="24dp"
        android:layout_marginEnd="16dp"
        android:background="@drawable/bg_form"
        android:elevation="8dp"
        android:orientation="vertical"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/header">

        <LinearLayout
            android:id="@+id/linearLayout5"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="14dp"
            android:gravity="center_horizontal"
            android:isScrollContainer="true"
            android:orientation="horizontal"
            android:paddingStart="10dp"
            android:paddingEnd="10dp">

            <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:orientation="vertical">

                <TextView
                    android:id="@+id/textView2"
                    android:layout_width="match_parent"
```

```
                    android:layout_height="48dp"
                    android:layout_weight="1"
                      android:fontFamily="@font/didact_gothic"
                      android:gravity="center_vertical"
                      android:text="Buat Tugas :"
                      android:textColor="#EDEDED"
                      android:textSize="20sp" />

                  <TextView
                      android:id="@+id/textView6"
                      android:layout_width="match_parent"
                      android:layout_height="48dp"
                      android:fontFamily="@font/didact_gothic"
                      android:gravity="center_vertical"
                      android:text="Nama Karyawan"
                      android:textColor="#EDEDED"
                      android:textSize="16sp" />

                  <TextView
                      android:id="@+id/textView9"
                      android:layout_width="match_parent"
                      android:layout_height="48dp"
                      android:fontFamily="@font/didact_gothic"
                      android:gravity="center_vertical"
                      android:text="Jenis Tugas"
                      android:textColor="#EDEDED"
                      android:textSize="16sp" />

                  <TextView
                      android:id="@+id/textView3"
                      android:layout_width="match_parent"
                      android:layout_height="48dp"
                      android:fontFamily="@font/didact_gothic"
                      android:gravity="center_vertical"
                      android:text="Nama \nPerusahaan"
                      android:textColor="#EDEDED"
                      android:textSize="16sp" />

                  <TextView
                      android:id="@+id/textView10"
                      android:layout_width="match_parent"
                      android:layout_height="48dp"
                      android:fontFamily="@font/didact_gothic"
                      android:gravity="center_vertical"
                      android:text="Alamat"
                      android:textColor="#EDEDED"
                      android:textSize="16sp" />

              </LinearLayout>

              <LinearLayout
                  android:layout_width="wrap_content"
```

```
                android:layout_height="match_parent"
                android:orientation="vertical"
                android:paddingStart="8dp"
                android:paddingEnd="8dp">

                <EditText
                    android:id="@+id/et_date"
                    android:layout_width="match_parent"
                    android:layout_height="48dp"
                    android:layout_marginRight="100dp"
                    android:layout_weight="1"
                    android:backgroundTint="#D1D1D1"
                    android:editable="false"
                    android:ems="10"
                    android:fontFamily="@font/didact_gothic"
                    android:hint="Tanggal . . ."
                    android:inputType="date"
                    android:textColorHint="#D1D1D1"
                    android:textSize="16sp" />

                <EditText
                    android:id="@+id/et_nama"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:backgroundTint="#D1D1D1"
                    android:ems="10"
                    android:fontFamily="@font/didact_gothic"
                    android:hint="Masukkan Nama Karyawan"
                    android:inputType="textPersonName"
                    android:minHeight="48dp"
                    android:singleLine="true"
                    android:textColor="#EDEDED"
                    android:textColorHint="#D1D1D1"
                    android:textSize="16sp" />

                <EditText
                    android:id="@+id/et_tugas"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:backgroundTint="#D1D1D1"
                    android:editable="false"
                    android:ems="10"
                    android:fontFamily="@font/didact_gothic"
                    android:hint="Jenis Tugas . . ."
                    android:inputType="textPersonName"
                    android:minHeight="48dp"
                    android:singleLine="true"
                    android:textColor="#EDEDED"
                    android:textColorHint="#D1D1D1"
                    android:textSize="16sp" />

                <EditText
```

```
                    android:id="@+id/et_perusahaan"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:backgroundTint="#D1D1D1"
                    android:ems="10"
                    android:fontFamily="@font/didact_gothic"
                    android:hint="Masukkan Nama Perusahaan"
                    android:inputType="textPersonName"
                    android:minHeight="48dp"
                    android:singleLine="true"
                    android:textColor="#EDEDED"
                    android:textColorHint="#D1D1D1"
                    android:textSize="16sp" />

                <EditText
                    android:id="@+id/et_alamat"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:backgroundTint="#D1D1D1"
                    android:ems="10"
                    android:fontFamily="@font/didact_gothic"
                    android:hint="Alamat"
                    android:inputType="textMultiLine"
                    android:minHeight="48dp"
                    android:textColor="#EDEDED"
                    android:textColorHint="#D1D1D1" />
            </LinearLayout>

        </LinearLayout>

        <Button
            android:id="@+id/btn_input_tugas"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginStart="50dp"
            android:layout_marginTop="25dp"
            android:layout_marginEnd="50dp"
            android:layout_marginBottom="20dp"
            android:background="@drawable/border_radius_all"
            android:fontFamily="@font/didact_gothic"
            android:padding="10dp"
            android:text="INPUT TUGAS"
            android:textSize="20sp" />

    </LinearLayout>

</androidx.constraintlayout.widget.ConstraintLayout>

```

***activity*_main_pantau.xml**

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#FFFFFF"
8.     tools:context=".MainPantau">
9.
10.    <fragment
11.        android:id="@+id/map_all"
12.
   android:name="com.google.android.gms.maps.SupportMapFragment"
13.        android:layout_width="match_parent"
14.        android:layout_height="match_parent"
15.        app:layout_constraintBottom_toBottomOf="parent"
16.        app:layout_constraintEnd_toEndOf="parent"
17.        app:layout_constraintHorizontal_bias="1.0"
18.        app:layout_constraintStart_toStartOf="parent"
19.        app:layout_constraintTop_toTopOf="parent" />
20.
21.    <LinearLayout
22.        android:layout_width="match_parent"
23.        android:layout_height="wrap_content"
24.        android:layout_marginStart="50dp"
25.        android:layout_marginTop="24dp"
26.        android:layout_marginEnd="50dp"
27.        android:background="@drawable/bg_search"
28.        android:orientation="vertical"
29.        android:padding="10dp"
30.        app:layout_constraintEnd_toEndOf="parent"
31.        app:layout_constraintStart_toStartOf="parent"
32.        app:layout_constraintTop_toTopOf="parent">
33.
34.        <EditText
35.            android:id="@+id/et_cari_nama"
36.            android:layout_width="match_parent"
37.            android:layout_height="48dp"
38.            android:backgroundTint="#7E7A7A"
39.            android:ems="10"
40.            android:fontFamily="@font/didact_gothic"
41.            android:hint="Cari nama karyawan"
42.            android:inputType="textPersonName"
43.            android:textColor="@color/black"
44.            android:textColorHighlight="#8C8C8C"
45.            android:textColorHint="#A5A2A2" />
46.
47.        <EditText
```

```
48.             android:id="@+id/et_cari_tgl"
49.             android:layout_width="match_parent"
50.             android:layout_height="wrap_content"
51.             android:backgroundTint="#7E7A7A"
52.             android:ems="10"
53.             android:enabled="true"
54.             android:focusableInTouchMode="false"
55.             android:fontFamily="@font/didact_gothic"
56.             android:hint="Tanggal Pencarian"
57.             android:minHeight="48dp"
58.             android:textColor="@color/black"
59.             android:textColorHint="#A5A2A2" />
60.
61.         <Button
62.             android:id="@+id/btn_cari"
63.             android:layout_width="wrap_content"
64.             android:layout_height="wrap_content"
65.             android:layout_gravity="center"
66.             android:backgroundTint="#1973FF"
67.             android:text="Cari" />
68.
69.     </LinearLayout>
70.
71. </androidx.constraintlayout.widget.ConstraintLayout>
72.
```

***activity*_main_user.xml**

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#ffffff"
8.     tools:context=".MainUser">
9.
10.    <androidx.constraintlayout.widget.ConstraintLayout
11.        android:id="@+id/header_3"
12.        android:layout_width="match_parent"
13.        android:layout_height="wrap_content"
14.        android:background="@drawable/border_radius_header"
15.        android:paddingBottom="10dp"
16.        app:layout_constraintEnd_toEndOf="parent"
17.        app:layout_constraintStart_toStartOf="parent"
18.        app:layout_constraintTop_toTopOf="parent">
19.
20.        <TextView
21.            android:id="@+id/textView"
22.            android:layout_width="wrap_content"
23.            android:layout_height="wrap_content"
24.            android:layout_marginStart="16dp"
25.            android:layout_marginTop="8dp"
26.            android:text="SIL Track v1.0"
27.            android:textColor="#4A4A4A"
28.            app:layout_constraintStart_toStartOf="parent"
29.            app:layout_constraintTop_toTopOf="parent" />
30.
31.        <ImageView
32.            android:id="@+id/imageView"
33.            android:layout_width="70dp"
34.            android:layout_height="70dp"
35.            android:layout_marginStart="16dp"
36.            android:layout_marginTop="16dp"
37.            android:background="@drawable/border_radius_logo"
38.            android:padding="10dp"
39.            app:layout_constraintStart_toStartOf="parent"
40.            app:layout_constraintTop_toBottomOf="@+id/textView"
41.            app:srcCompat="@drawable/receptionist" />
42.
43.        <TextView
44.            android:id="@+id/tv_nama_karyawan"
45.            android:layout_width="wrap_content"
46.            android:layout_height="wrap_content"
47.            android:layout_marginStart="18dp"
48.            android:layout_marginTop="40dp"
```

```
            android:hint="nama_karyawan"
            android:textColor="#121010"
            android:textColorHint="#737070"
            android:textSize="30sp"
            app:layout_constraintStart_toEndOf="@+id/imageView"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/tv_jabatan"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="18dp"
            android:hint="jabatan"
            android:textColor="#818181"
            android:textColorHint="#818181"
            android:textSize="22sp"
            app:layout_constraintStart_toEndOf="@+id/imageView"
   app:layout_constraintTop_toBottomOf="@+id/tv_nama_karyawan" />

        <com.google.android.material.tabs.TabLayout
            android:id="@+id/tl_menu"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginStart="28dp"
            android:layout_marginTop="16dp"
            android:layout_marginEnd="28dp"
            android:layout_marginBottom="30dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/tv_jabatan"
            app:tabIndicatorColor="#ffc246"
            app:tabInlineLabel="true">

            <com.google.android.material.tabs.TabItem
                android:id="@+id/item_informasi"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:icon="@drawable/baseline_assignment_24"
                android:text="Informasi" />

            <com.google.android.material.tabs.TabItem
                android:id="@+id/item_lokasi"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:icon="@drawable/baseline_map_24"
                android:text="Lokasi Saya" />

        </com.google.android.material.tabs.TabLayout>

    </androidx.constraintlayout.widget.ConstraintLayout>
```

```
100.
101.     <androidx.viewpager.widget.ViewPager
102.         android:id="@+id/vp"
103.         android:layout_width="match_parent"
104.         android:layout_height="match_parent"
105.         android:layout_marginStart="14dp"
106.         android:layout_marginTop="174dp"
107.         android:layout_marginEnd="14dp"
108.         android:background="@drawable/border_radius_body"
109.         android:visibility="visible"
110.         app:layout_constraintEnd_toEndOf="parent"
111.         app:layout_constraintStart_toStartOf="parent"
112.         app:layout_constraintTop_toBottomOf="@id/header_3"
113.         tools:ignore="SpeakableTextPresentCheck"
114.         tools:visibility="visible" >
115.
116.     </androidx.viewpager.widget.ViewPager>
117.
118.
    <com.google.android.material.floatingactionbutton.FloatingActionB
    utton
119.         android:id="@+id/float_bt"
120.         android:layout_width="wrap_content"
121.         android:layout_height="wrap_content"
122.         android:layout_marginBottom="32dp"
123.         android:clickable="true"
124.         app:layout_constraintBottom_toBottomOf="parent"
125.         app:layout_constraintEnd_toEndOf="parent"
126.         app:layout_constraintHorizontal_bias="0.9"
127.         app:layout_constraintStart_toStartOf="parent"
128.         app:srcCompat="@drawable/business_center"
129.         tools:ignore="SpeakableTextPresentCheck" />
130.
131.     <ProgressBar
132.         android:id="@+id/progressBar"
133.         style="?android:attr/progressBarStyle"
134.         android:layout_width="0dp"
135.         android:layout_height="wrap_content"
136.         android:visibility="invisible"
137.         app:layout_constraintBottom_toBottomOf="parent"
138.         app:layout_constraintEnd_toEndOf="parent"
139.         app:layout_constraintStart_toStartOf="parent"
140.         app:layout_constraintTop_toTopOf="parent" />
141.
142. </androidx.constraintlayout.widget.ConstraintLayout>
143.
```

***activity*_splash_screen.xml**

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="#FFFFFF"
8.     tools:context=".SplashScreen">
9.
10.    <ImageView
11.        android:id="@+id/img_logo_apk"
12.        android:layout_width="100dp"
13.        android:layout_height="100dp"
14.        app:layout_constraintBottom_toBottomOf="parent"
15.        app:layout_constraintEnd_toEndOf="parent"
16.        app:layout_constraintHorizontal_bias="0.5"
17.        app:layout_constraintStart_toStartOf="parent"
18.        app:layout_constraintTop_toTopOf="parent"
19.        app:layout_constraintVertical_bias="0.5"
20.        app:srcCompat="@drawable/logo_tracking" />
21.
22.</androidx.constraintlayout.widget.ConstraintLayout>
23.
```

fragment_daftar_tugas.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/coordinator"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/border_radius_body"
    android:nestedScrollingEnabled="true"
    android:overScrollMode="ifContentScrolls"
    android:paddingBottom="20dp"
    tools:context=".fragment_daftar_tugas">

    <TableLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal|top"
        android:layout_marginTop="30dp"
        android:background="@drawable/border_radius_table">

        <TableRow
            android:id="@+id/tabel_jam"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/judul_masuk"
                android:layout_width="wrap_content"
                android:layout_height="48dp"
                android:clickable="true"
                android:focusable="true"
                android:fontFamily="@font/didact_gothic"
                android:padding="10dp"
                android:text="Jam Masuk"
                android:textAlignment="center"
                android:textColor="#393939"
                android:textSize="16sp" />

            <TextView
                android:id="@+id/judul_keluar"
                android:layout_width="wrap_content"
                android:layout_height="48dp"
                android:clickable="true"
                android:focusable="true"
                android:fontFamily="@font/didact_gothic"
                android:padding="10dp"
                android:text="Jam Keluar"
                android:textAlignment="center"
```

```
                android:textColor="#393939"
                android:textSize="16sp" />

        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/tv_masuk"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:fontFamily="@font/didact_gothic"
                android:padding="10dp"
                android:text="- - . - -"
                android:textAlignment="center"
                android:textColor="#393939"
                android:textSize="16sp" />

            <TextView
                android:id="@+id/tv_keluar"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:fontFamily="@font/didact_gothic"
                android:padding="10dp"
                android:text="- - . - -"
                android:textAlignment="center"
                android:textColor="#393939"
                android:textSize="16sp" />
        </TableRow>

    </TableLayout>

    <TextView
        android:id="@+id/textView29"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerInParent="false"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="120dp"
        android:fontFamily="@font/didact_gothic"
        android:text="Berikut tugas anda hari ini"
        android:textColor="#393939"
        android:textSize="20sp" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/rv_daftar_tugas"
        android:layout_width="match_parent"
        android:layout_height="584dp"
          android:layout_below="@id/textView29"
```

```
101.          android:layout_alignParentBottom="false"
102.          android:layout_marginTop="150dp"
103.          android:isScrollContainer="true"
104.          android:paddingStart="10dp"
105.          android:paddingBottom="15dp" />
106.
107.      <ImageView
108.          android:id="@+id/img_selesai"
109.          android:layout_width="100dp"
110.          android:layout_height="100dp"
111.          android:layout_alignParentLeft="false"
112.          android:layout_alignParentTop="false"
113.          android:layout_alignParentRight="false"
114.          android:layout_alignParentBottom="false"
115.          android:layout_gravity="center_horizontal"
116.          android:layout_marginTop="200dp"
117.          android:src="@drawable/thumbs_up" />
118.
119. </androidx.coordinatorlayout.widget.CoordinatorLayout>
120.
```

fragment_map_user.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:tools="http://schemas.android.com/tools"
4.     android:layout_width="match_parent"
5.     android:layout_height="match_parent"
6.     android:background="@drawable/bg_map_radius"
7.     tools:context=".fragment_map_user">
8.
9.     <fragment
10.        android:id="@+id/map"
11.
   android:name="com.google.android.gms.maps.SupportMapFragment"
12.        android:layout_width="match_parent"
13.        android:layout_height="match_parent" />
14. </RelativeLayout>
15.
```

layout_recycle_view_tugas.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <androidx.cardview.widget.CardView
   xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:id="@+id/cv_tugas"
6.     android:layout_width="280dp"
7.     android:layout_height="wrap_content"
8.     android:layout_gravity="center|top"
9.     android:layout_marginStart="10dp"
10.    android:layout_marginTop="20dp"
11.    android:clickable="true"
12.    android:padding="10dp"
13.    app:cardBackgroundColor="#FFFFFF"
14.    app:cardCornerRadius="8dp"
15.    app:cardElevation="5dp"
16.    app:layout_constraintLeft_toLeftOf="parent">
17.
18.    <LinearLayout
19.        android:layout_width="match_parent"
20.        android:layout_height="wrap_content"
21.        android:layout_marginStart="10dp"
22.        android:layout_marginTop="5dp"
23.        android:background="#FFFFFF"
24.        android:orientation="vertical">
25.
26.        <TextView
27.            android:id="@+id/tv_judul_tugas"
28.            android:layout_width="match_parent"
29.            android:layout_height="wrap_content"
30.            android:layout_marginBottom="10dp"
31.            android:fontFamily="@font/didact_gothic"
32.            android:paddingEnd="10dp"
33.            android:textColor="#000000"
34.            android:textSize="20sp"
35.            android:textStyle="bold" />
36.
37.        <TextView
38.            android:id="@+id/tv_nama_perusahaan"
39.            android:layout_width="match_parent"
40.            android:layout_height="wrap_content"
41.            android:layout_marginBottom="10dp"
42.            android:fontFamily="@font/didact_gothic"
43.            android:paddingEnd="10dp"
44.            android:textColor="#000000"
45.            android:textSize="16sp" />
46.
47.        <TextView
48.            android:id="@+id/tv_alamat"
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="10dp"
            android:fontFamily="@font/didact_gothic"
            android:paddingEnd="10dp"
            android:textColor="#000000"
            android:textSize="16sp"
            android:textStyle="italic" />
    </LinearLayout>
</androidx.cardview.widget.CardView>

```