

BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Teori Umum

2.1.1 Sistem Informasi

Menurut Kevin et al., (2016) secara umum, pengertian sistem informasi adalah suatu sistem yang menyediakan informasi untuk manajemen pengambilan keputusan atau kebijakan dan menjalankan operasional dari kombinasi orang-orang, teknologi informasi dan prosedur-prosedur yang terorganisasi. atau sistem informasi diartikan sebagai kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi untuk mendukung operasi dan manajemen. Sedangkan dalam arti luas, sistem informasi diartikan sebagai sistem informasi yang sering digunakan menurut kepada interaksi antara orang, proses, algoritmik, data dan teknologi.

Menurut Kevin et al., (2016), secara terpisah, pengertian sistem adalah kumpulan orang yang saling bekerja sama dengan ketentuan-ketentuan aturan yang sistematis dan terstruktur untuk membentuk satu kesatuan melaksanakan suatu fungsi untuk mencapai tujuan. Sistem memiliki beberapa karakteristik atau sifat yang terdiri dari komponen sistem, batasan sistem, lingkungan luar sistem, penghubung sistem, masukan sistem, keluaran sistem, pengolahan sistem dan sasaran sistem. Sedangkan pengertian informasi adalah data yang diolah menjadi

lebih berguna dan berarti bagi penerimanya dan untuk mengurangi ketidakpastian dalam proses pengambilan keputusan mengenai suatu keadaan.

2.1.1.1 Fungsi Sistem Informasi

Untuk meningkatkan aksesibilitas data yang ada secara efektif dan efisien kepada pengguna, tanpa dengan prantara sistem informasi. Memperbaiki produktivitas aplikasi pengembangan dan pemeliharaan sistem. Menjamin tersedianya kualitas dan keterampilan dalam memanfaatkan sistem informasi secara kritis. Mengidentifikasi kebutuhan mengenai keterampilan pendukung sistem informasi. Mengantisipasi dan memahami akan konsekuensi ekonomi. Menetapkan investasi yang akan diarahkan pada sistem informasi. Mengembangkan proses perencanaan yang efektif. (Kevin et al., 2016)

2.1.1.2 Komponen Sistem Informasi

Menurut Kevin et al., (2016), Komponen-komponen dari sistem informasi adalah sebagai berikut ;

1. Komponen *input* adalah data yang masuk ke dalam sistem informasi.
2. Komponen model adalah kombinasi prosedur, logika dan model matematika yang memproses data yang tersimpan di basis data dengan cara yang sudah di tentukan untuk menghasilkan keluaran yang diinginkan.

3. Komponen *output* adalah hasil informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.
4. Komponen teknologi adalah alat dalam sistem informasi, teknologi digunakan dalam menerima *input*, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan *output* dan memantau pengendalian sistem.
5. Komponen basis data adalah kumpulan data yang saling berhubungan yang tersimpan di dalam komputer dengan menggunakan *software database*.
6. Komponen kontrol adalah komponen yang mengendalikan gangguan terhadap sistem informasi.

2.1.2 Sistem Informasi Geografis

Menurut R & Yapie, (2013), sistem informasi geografis (bahasa inggris : *geographic information system*) adalah sistem informasi khusus yang mengelola data yang memiliki informasi spasial (bereferensi keruangan), Atau dalam arti yang lebih sempit adalah *system computer* yang memiliki kemampuan untuk membangun, menyimpan, mengelola, menampilkan data spasial maupun non spasial yang bereferensi geografis. Data digital geografis di organisir menjadi 2 bagian, yaitu (R & Yapie, 2013):

1. Data Spasial

Data yang menyimpan komponen-komponen permukaan bumi, seperti: jalan, pemukiman, jenis penggunaan tanah, jenis tanah, dan lain-lain. Model data spasial dibedakan menjadi dua, yaitu: model data raster (Model data yang menampilkan, menempatkan, dan menyimpan spasial dengan menggunakan struktur matriks atau *pixel-pixel* yang membentuk grid) dan model data vector (Model data yang menampilkan, menempatkan, dan menyimpan data spasial dengan menggunakan titik, garisgaris, atau kurva atau polygon beserta atribut-atributnya).

2. Data Non Spasial (Tabular/atribut)

Model data non spasial adalah data yang menyimpan atribut dari kenampakan-kenampakan permukaan bumi tersebut. Misalnya tanah yang memiliki atribut tekstur, kedalaman, struktur pH, dan lainnya.

2.1.2.1 Mobile GIS (*Geographic Information System*)

Teknologi GIS (*Geographic Information System*) mengalami perkembangan yang sangat pesat. Diantaranya adalah Mobile GIS dimana GIS yang tadinya hanya digunakan di dalam lingkungan kantor menjadi semakin fleksibel dan mampu digunakan di luar kantor secara mobile. *Mobile Geographic Information System* dapat digunakan untuk menangkap, menyimpan, *update*, manipulasi, analisa dan menampilkan informasi geografi secara mudah. *Mobile GIS* mengintegrasikan salah satu atau lebih teknologi berikut (R & Yapie, 2013):

1. Perangkat *Mobile*.
2. *Global Positioning System* (GPS).
3. *Wireless communication* untuk mengakses *internet* GIS.

Biasanya proses mengumpulkan dan *editing* data menghabiskan banyak waktu dan sering terjadi kesalahan. Data *geographic* biasanya diperoleh dari lapangan melalui lembaran-lembaran peta. Proses *editing* dilakukan secara manual dengan cara memasukkannya kedalam database GIS. Hasilnya data menjadi tidak *up-to-date* dan tidak akurat. (R & Yapie, 2013)

Teknologi *Mobile* GIS memungkinkan GIS dapat langsung diimplementasikan dilapangan sebagai peta digital, mobile computer sehingga informasi dapat di tambahkan secara *real time* ke database dan aplikasinya, mempercepat analisis, *display*, dan pengambilan keputusan dengan data yang *up-to-date* dan akurat. (R & Yapie, 2013)

2.1.3. *Unified Modelling Language (UML)*

Menurut Berbudhi Rachman & Februariyanti, (2013), *Unified Modelling Language (UML)* adalah bahasa spesifikasi standard untuk mendokumentasikan, menspesifikasikan dan membangun sistem perangkat lunak. UML tidak didasarkan pada bahasa pemrograman tertentu. Standar spesifikasi UML dijadikan standar de facto oleh *OMG (Object Management Group)* pada tahun 2007. UML yang berorientasikan object mempunyai beberapa notasi standar.

Spesifikasi ini menjadi populer dan standar karena sebelum adanya UML, telah ada berbagai macam spesifikasi yang berbeda. Hal ini menyulitkan komunikasi antara pengembang perangkat lunak. Untuk itu, beberapa pengembang spesifikasi yang sangat berpengaruh berkumpul untuk membuat standar baru. UML dirintis oleh Grady Booch, James Rumbaugh pada tahun 1994. (Berbudhi Rachman & Februariyanti, 2013)

2.1.3.1 Use Case Diagram

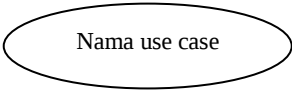
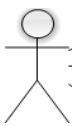
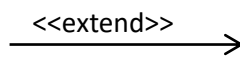
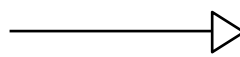
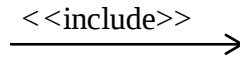
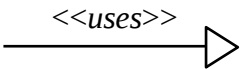
Menurut Rosa A.S.,(2015), Diagram *use case* merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimple mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case* (Rosa A.S., 2015);

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor itu orang, tapi actor belum tentu merupakan orang.

2. Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukaran pesan antara unit atau aktor

Tabel 2.1. Penjelasan simbol-simbol *Use Case Diagram*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama <i>use case</i> .
Aktor / <i>actor</i>  Nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
Asosiasi / <i>association</i> 	Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor
Relasi 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu; mirip dengan prinsip <i>inheritance</i> pada pemrograman berorientasi objek
Generalisasi / <i>generalization</i> 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Menggunakan / <i>include</i> / <i>uses</i>  	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>use case</i> ini Ada dua sudut pandang yang cukup besar mengenai <i>include</i> di <i>use case</i> : 1. <i>Include</i> berarti <i>use case</i> yang ditambahkan akan selalu dipanggil saat <i>use case</i> tambahan dijalankan 2. <i>Include</i> berarti <i>use case</i> yang tambahan akan selalu melakukan pengecekan apakah <i>use case</i> yang ditambahkan telah dijalankan sebelum <i>use case</i> tambahan dijalankan. Kedua interpretasi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan

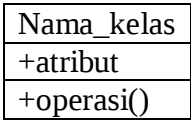
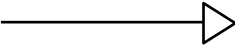
Sumber : (Rosa A.S., 2015)

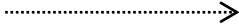
2.1.3.2 Diagram Kelas (*Class Diagram*)

Class diagram menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Kelas memiliki apa yang disebut atribut dan metode atau operasi (Djafar, 2015).

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Tabel 2.2. Penjelasan simbol-simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka / <i>interface</i>  nama_interface	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
Asosiasi / <i>association</i> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Asosiasi beralih / <i>directed association</i> 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum – khusus)

Simbol	Deskripsi
Kebergantungan / <i>dependency</i> 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / <i>aggregation</i> 	Relasi antar kelas dengan makna semua-bagian (<i>whole-part</i>)

Sumber: (Rosa A.S., 2015)

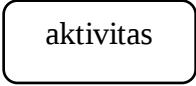
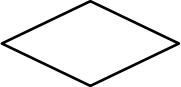
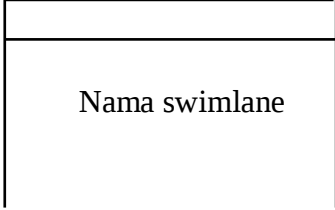
2.1.3.3 Diagram Aktifitas (*Activity Diagram*)

Menurut Nama et al., (2015), Diagram aktivitas digunakan untuk menggambarkan perilaku aliran kontrol atau arus objek yang mungkin terjadi pada sistem. Semua kegiatan dikoordinasikan oleh beberapa model Itu bisa dimulai karena tindakan lain sudah selesai dijalankan, objek dan data tersedia, atau beberapa pemicu eksternal lainnya yang menyebabkan aliran terjadi.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut (Rosa A.S., 2015) :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Tabel 2.3. Penjelasan simbol-simbol *Activity Diagram*

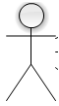
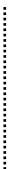
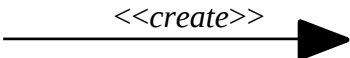
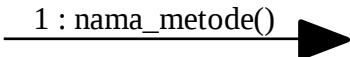
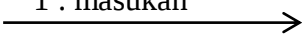
Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / <i>decision</i> 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / <i>join</i> 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
<i>Swimlane</i> 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

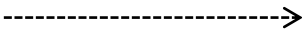
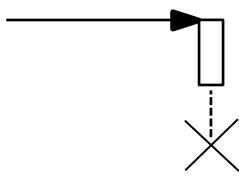
Sumber: (Rosa A.S., 2015)

2.1.3.4. *Sequence Diagram*

Menurut Djafar (2015), *Sequence diagram* yang dirancang menjelaskan secara detail urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *use case*, interaksi yang terjadi antar kelas, operasi apa saja yang terlibat, urutan antara operasi dan informasi yang perlu dilakukan.

Tabel 2.4. Penjelasan Squence Diagram

Simbol	Deskripsi
Aktor  Nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri
Garis hidup / <i>lifeline</i> 	Menyatakan kehidupan suatu objek
Objek <u>nama objek : nama kelas</u>	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
pesan tipe <i>creat</i> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
pesan tipe <i>call</i> 	Menyatakan suatu objek operasi memanggil / metode yang ada pada objek lain atau dirinya sendiri.
Pesan tipe <i>send</i> 	Menyatakan bahwa suatu objek mengirim data / masukan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim

Simbol	Deskripsi
Pesan tipe <i>return</i> 1 : keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe <i>destroy</i> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada <i>create</i> maka ada <i>destroy</i>

Sumber: (Rosa A.S., 2015)

2.2. Tinjauan Teori khusus

2.2.1 Pariwisata

Menurut Tri Handoyo, M. Kom dan Wahyu Priyoatmoko (2014), Pariwisata adalah salah satu jenis industri baru yang mampu mempercepat pertumbuhan ekonomi dan penyediaan lapangan pekerjaan, peningkatan penghasilan, standar hidup serta menstimulasi sektor-sektor produktif lainnya.

Menurut Tri Handoyo, M. Kom dan Wahyu Priyoatmoko (2014), Kepariwisata adalah hakikatnya dari perlawatan serta masa tinggal pengunjung-pengunjung asing ke suatu Negara atau tempat, sepanjang tinggalnya itu tidak mengakibatkan penghunian, berdomisili, atau suatu keadaan tinggal, menetap dan tidak pula mengakibatkan suatu hubungan dengan pihak tertentu yang bersifat *employement* (bekerja mencari nafkah).

2.2.1.1 Jenis Pariwisata

Disamping bentuknya, jenis pariwisata perlu pula dibicarakan di sini untuk menyusun statistik atau data-data penelitian dan peninjauan yang lebih akurat dalam bidang ini. Jenis-jenis yang telah dikenal saat ini antara lain (Tri Handoyo, M. Kom dan Wahyu Priyoatmoko, 2014) :

1. Wisata Alam

Wisata ini diselenggarakan oleh agen atau biro perjalanan yang mengkhususkan usaha-usaha dengan mengatur wisata ke tempat atau daerah cagar alam, taman lindung, hutan daerah pegunungan, air terjun dan sebagainya yang kelestariannya dilindungi oleh undang-undang.

2. Wisata Sejarah atau Budaya

Wisata yang dilakukan atas dasar keinginan, untuk memperluas pandangan hidup seseorang dengan jalan mengadakan kunjungan atau peninjauan ke tempat lain atau luar negeri, mempelajari keadaan rakyat, kebiasaan adat istiadat mereka, cara hidup mereka, budaya sejarah, dan seni mereka.

3. Wisata Petualang atau Wisata Minat dan Khusus

Wisata dengan petualang dengan minat khusus seperti masuk ke hutan belantara, pendakian gunung, mendaki tebing yang terjal, terjun ke air terjun yang sangat curam dan sebagainya.

4. Wisata Agrowisata

Wisata ini adalah pengorganisasian perjalanan yang dilakukan oleh proyek-proyek pertanian, perkebunan, ladang pembibitan dan sebagainya dimana wisatawan dapat mengadakan kunjungan dan peninjauan untuk tujuan studi maupun melihat-lihat sambil menikmati segarnya jenis sayur, kebun dan lain sebagainya.

5. Wisata Industri

Perjalanan yang dilakukan oleh rombongan seorang ke suatu kompleks atau daerah perindustrian dimana terdapat pabrik-pabrik atau bengkel-bengkel besar dengan maksud mengadakan peninjauan atau penelitian.

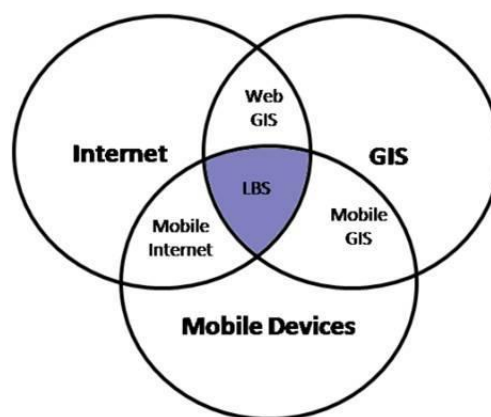
6. Wisata Religi

Wisata ini dikaitkan dengan wisata agama dan kepercayaan umat atau sekelompok dalam masyarakat yang berupa mengunjungi makam keramat, masjid, gereja, wihara, pura dan tempat-tempat dengan simbol agama.

2.2.2 *Location Based Services (LBS)*

Menurut Isworo Nugroho (2012), *Location Based Services* adalah layanan informasi yang dapat diakses menggunakan piranti *mobile* melalui jaringan Internet dan seluler serta memanfaatkan kemampuan penunjuk lokasi pada piranti *mobile*. *Location Based Systems* melakukan komunikasi dan interaksi dua arah. *Location Based Services* dapat digambarkan sebagai suatu layanan yang berada

pada pertemuan tiga teknologi yaitu : *Geographic Information System*, *Internet Service*, dan *Mobile Devices*. Hal ini dapat dilihat pada gambar dibawah ini (Isworo Nugroho, 2012):



Gambar 2.1. Layanan *Location Based Services*

Sumber: (Isworo Nugroho, 2012)

Teknologi *Location Based service* (LBS) merupakan salah satu bagian dari implementasi *mobile GIS* yang lebih cenderung memberikan fungsi terapan sehari-hari seperti menampilkan direktori kota, navigasi kendaraan, pencarian alamat serta jejaring sosial dibanding fungsionalitas pada teknologi GIS populer untuk *Field Based GIS*. Dua unsur utama LBS adalah (R & Yapie, 2013) :

1. *Location Manager (API Maps)*: Menyediakan *tools/source* untuk LBS, *Application Programming Interface (API) Maps* menyediakan fasilitas untuk menampilkan, memanipulasi peta beserta *feature* lainnya seperti tampilan satelit, *street* (jalan), maupun gabungannya. Paket ini berada pada *com.google.android.maps*.
2. *Location Providers (API Location)*: Menyediakan teknologi pencarian lokasi yang digunakan oleh *device/perangkat*. *API Location* berhubungan

dengan data GPS (*Global Positioning System*) dan data lokasi *real-time*. API Location berada pada paket *android* yaitu *android.location*. Dengan *location manager*, kita dapat menentukan lokasi kita saat ini dan rute menuju tempat tertentu.

Layanan Berbasis Lokasi terdiri 5 komponen utama yaitu (Berbudhi Rachman & Februariyanti, 2013):



Gambar 2.2 Komponen utama *Location Based Service*

Sumber : (Anwar, Nugroho, & Lestariningsih, 2013)

1. Piranti *Mobile*

Piranti *Mobile* adalah salah satu komponen penting dalam LBS. Piranti ini berfungsi sebagai alat bantu (*tool*) bagi pengguna untuk meminta informasi. Hasil dari informasi yang diminta dapat berupa teks, suara, gambar dan lain sebagainya. Piranti *mobile* yang dapat digunakan bisa berupa PDA, *smartphone*, *laptop* dan PC. Selain itu, piranti *mobile* dapat juga berfungsi sebagai alat navigasi berbasis *GPS*.

2. Jaringan Komunikasi

Komponen kedua adalah jaringan komunikasi. Komponen ini berfungsi sebagai jalur penghubung yang dapat mengirim data-data yang dikirim oleh pengguna dari piranti *mobile*-nya untuk kemudian dikirimkan ke penyedia layanan dan kemudian hasil permintaan tersebut dikirimkan kembali oleh penyedia layanan kepadapengguna.

3. Komponen *Positioning* (Penunjuk Posisi/Lokasi)

Komponen selanjutnya adalah Komponen *Positioning* (Penunjuk Posisi/Lokasi). Setiap layanan yang diberikan oleh penyedia layanan biasanya akan berdasarkan pada posisi pengguna yang meminta layanan tersebut. Oleh karena itu diperlukan komponen yang berfungsi sebagai pengolah/pemroses yang akan menentukan posisi pengguna layanan saat itu. Posisi pengguna tersebut bisa didapatkan melalui jaringan komunikasi *mobile* atau juga menggunakan *Global Positioning System* (GPS).

4. Penyedia Layanan/Aplikasi

Penyedia layanan merupakan komponen LBS yang memberikan berbagai macam layanan yang bisa digunakan oleh pengguna. Sebagai contoh ketika pengguna meminta layanan agar bisa tahu posisinya saat itu, maka aplikasi dan penyedia layanan langsung memproses permintaan tersebut, mulai dari menghitung dan menentukan posisi pengguna, menemukan rute jalan, dan masih banyak lagi yang lainnya.

5. *Data and Content Provider*

Penyedia layanan tidak selalu menyimpan semua data yang dibutuhkan yang bisa diakses oleh pengguna. Untuk itu, data dapat diminta dari content provider.

Memanfaatkan LBS pada dasarnya melakukan aktivitas sebagai berikut : menentukan posisi pengguna terhadap seseorang atau objek lain (***locating***), mencari seseorang, objek atau suatu kejadian (***searching***) dan arah menuju lokasi yang dituju (***navigating***), informasi mengenai suatu objek atau kejadian (***identifying***) atau mencari kejadian yang paling dekat dengan posisi pengguna (***checking***). (Isworo Nugroho, 2012)

2.2.2.1 Cara Kerja Location Based Service

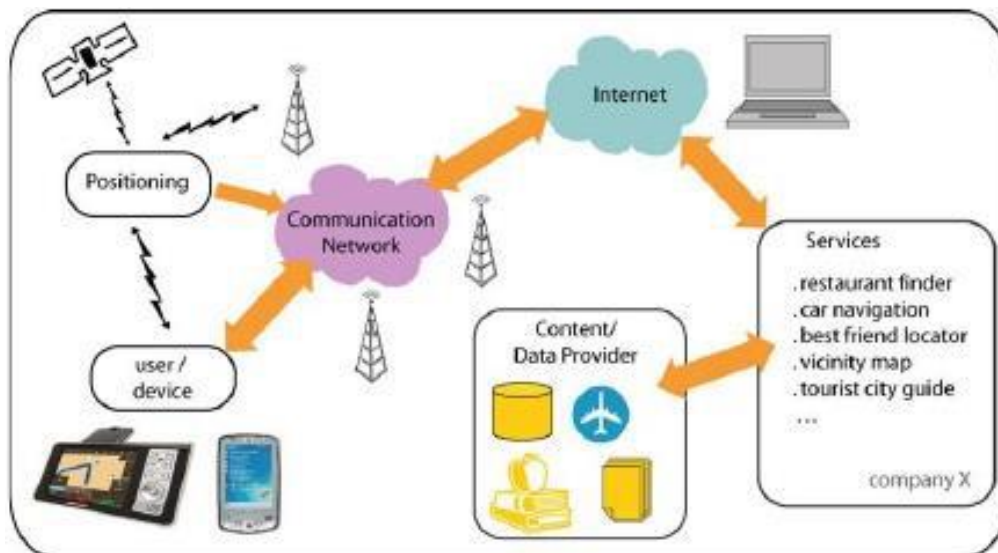
Untuk menggambarkan cara kerja LBS, anggaplah aplikasi LBS akan mencari informasi mengenai lokasi restaurant yang berada di sekitar posisi kita sekarang. (Djafar, 2015)

1. Anggaplah sekarang fungsi pencarian telah diaktifkan, posisi pengguna sebenarnya dari perangkat mobile diperoleh dari *positioning service*. Hal ini dapat dilakukan baik oleh perangkat menggunakan GPS sendiri atau layanan posisi jaringan yang berasal dari provider (Cell Tower). Setelah itu perangkat *mobile* pengguna mengirimkan permintaan informasi, yang berisi tujuan untuk mencari dan mengirimkan posisi melalui jaringan komunikasi ke *gateway* telekomunikasi.

2. *Gateway* memiliki tugas untuk bertukar pesan di antara jaringan komunikasi selular dan internet. Oleh karena itu, dia tahu alamat *web* dari beberapa aplikasi *server* dan rute permintaan ke spesifik *server* tertentu. *Gateway* akan menyimpan juga informasi tentang perangkat *mobile* yang telah meminta informasi.
3. Aplikasi *server* membaca permintaan dan mengaktifkan layanan yang terkait. Dalam kasus ini layanan pencarian.
4. Sekarang, *service* menganalisis lagi pesan dan memutuskan mana informasi tambahan selain dari kriteria pencarian (restaurant berserta padang) dan posisi pengguna diperlukan untuk menjawab permintaan pengguna. Dalam kasus ini *service* akan menemukan bahwa pengguna membutuhkan informasi tentang restaurant dari *database yellow pages* pada wilayah tertentu dan kemudian *service* tersebut akan meminta penyedia data untuk memberikan data tersebut.
5. Selanjutnya *service* akan menemukan bahwa informasi tentang jalan, jarak dan cara yang diperlukan untuk memeriksa apakah restaurant dapat dicapai.
6. Setelah sekarang semua informasi *service* akan melakukan *buffer* spasial dan *query routing* untuk mendapatkan beberapa restaurant terdekat. Setelah menghitung daftar restaurant, hasil dikirim kembali ke pengguna melalui internet, *gateway* dan jaringan *mobile*.

Informasi mengenai restaurant sekarang akan disampaikan kepada pengguna baik sebagai daftar teks (disusun berdasarkan jarak) atau ditampilkan dalam peta.

Setelah itu pengguna dapat meminta informasi lebih lanjut tentang restaurant (misalnya menu dan harga), yang mengaktifkan jenis layanan berbeda. Akhirnya jika pengguna memilih restaurant tertentu dia bisa meminta rute ke restaurant itu.



Gambar 2.3. Cara Kerja *Location Based Service*

Sumber : (Djafar, 2015)

2.2.3 *Global Positioning System (GPS)*

Menurut Santoso et al., (2016), *Global Positioning System (GPS)* merupakan suatu kumpulan satelit dan *system control* yang memungkinkan sebuah penerima GPS untuk mendapatkan lokasinya dipermukaan bumi 24 jam sehari. Sistem ini menggunakan sejumlah satelit yang berada di orbit bumi, yang memancarkan sinyal ke bumi dan ditangkap oleh sebuah alat penerima. *Global Positioning System (GPS)* adalah sistem untuk menentukan posisi di permukaan bumi dengan bantuan sinkronisasi sinyal satelit. Sistem ini menggunakan minimal 4 satelit yang mengirimkan sinyal gelombang mikro ke Bumi. Sinyal ini diterima

oleh alat penerima di permukaan, dan digunakan untuk menentukan posisi, kecepatan, arah, dan waktu.

Ada tiga bagian penting dari sistem ini, yaitu bagian (Santoso et al., 2016);

1. Bagian Kontrol

Berfungsi sebagai pengontrol, Setiap satelit dapat berada sedikit diluar orbit, sehingga bagian ini melacak orbit satelit, lokasi, ketinggian, dan kecepatan. Sinyal-sinyal dari satelit diterima oleh bagian kontrol, dikoreksi, dan dikirimkan kembali ke satelit. Koreksi data lokasi yang tepat dari satelit ini disebut dengan data ephemeris, yang nantinya akan di kirimkan kepada alat navigasi kita.

2. Bagian Angkasa

Bagian ini terdiri dari kumpulan satelit-satelit yang berada di orbit bumi, sekitar 12.000 mil diatas permukaan bumi. Kumpulan satelit-satelit ini diatur sedemikian rupa sehingga alat navigasi setiap saat dapat menerima paling sedikit sinyal dari empat buah satelit. Sinyal satelit ini dapat melewati awan, kaca, atau plastik, tetapi tidak dapat melewati gedung atau gunung. Satelit mempunyai jam atom, dan juga akan memancarkan informasi 'waktu/jam' ini. Data ini dipancarkan dengan kode 'pseudorandom'. Masing-masing satelit memiliki kodenya sendiri-sendiri. Nomor kode ini biasanya akan ditampilkan di alat navigasi, maka kita bisa melakukan identifikasi sinyal satelit yang sedang diterima alat tersebut. Data ini berguna bagi alat navigasi untuk mengukur jarak

antara alat navigasi dengan satelit, yang akan digunakan untuk mengukur koordinat lokasi.

3. Bagian Pengguna

Bagian ini terdiri dari alat navigasi yang digunakan. Satelit akan memancarkan data *almanak* dan *ephemeris* yang akan diterima oleh alat navigasi secara teratur. Data *almanak* berisikan perkiraan lokasi (*approximate location*) satelit yang dipancarkan terus menerus oleh satelit. Data *ephemeris* dipancarkan oleh satelit, dan valid untuk sekitar 4-6 jam. Untuk menunjukkan koordinat sebuah titik (dua dimensi), alat navigasi memerlukan paling sedikit sinyal dari 3 buah satelit. Untuk menunjukkan data ketinggian sebuah titik (tiga dimensi), diperlukan tambahan sinyal dari 1 buah satelit lagi. Dari sinyal-sinyal yang dipancarkan oleh kumpulan satelit tersebut, alat navigasi akan melakukan perhitungan-perhitungan, dan hasil akhirnya adalah koordinat posisi alat tersebut. Makin banyak jumlah sinyal satelit yang diterima oleh sebuah alat, akan membuat alat tersebut menghitung koordinat posisinya dengan lebih tepat.

2.2.4 Google Map API

Menurut Santoso et al., (2016), API atau *Application Programming Interface* merupakan suatu dokumentasi yang terdiri dari *interface*, fungsi, kelas, struktur dan sebagainya untuk membangun sebuah perangkat lunak. Dengan

adanya *API* ini, maka memudahkan programmer untuk “membongkar” suatu software untuk kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. *API* dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya yang memungkinkan programmer menggunakan system function. Proses ini dikelola melalui *operating system*. Keunggulan dari *API* ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi. Bahasa yang digunakan oleh *Google Map* yang terdiri dari *HTML*, *JavaScript* dan *AJAX* serta *XML*, memungkinkan untuk menampilkan peta *Google Map* di website lain. *Google* juga menyediakan layanan *Google Map API* yang memungkinkan para pengembang untuk mengintegrasikan *Google Map* ke dalam website masing-masing dengan menambahkan data point sendiri. Dengan menggunakan *Google Map API*, *Google Map* dapat ditampilkan pada web site eksternal. Agar aplikasi *Google Map* dapat muncul di website tertentu, diperlukan adanya *API key*. *API key* merupakan kode unik yang digenerasikan oleh *Google* untuk suatu website tertentu, agar *server Google Map* dapat mengenali (Davis, 2006).

2.2.5 *Android*

Menurut R & Yapie, (2013), *Android* adalah sistem operasi untuk *handphone* yang berbasis *Linux*. *Android* menyediakan *platform* terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, *Google Inc.* membeli *Android Inc.*

pendatang baru yang membuat peranti lunak untuk *handphone* . Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi.

Pada saat perilisan perdana *Android*, 5 November 2007, *Android* bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode *Android* di bawah lisensi Apache, sebuah lisensi perangkat *software* dan standar terbuka perangkat seluler. (R & Yapie, 2013).

Di dunia ini terdapat dua jenis distributor sistem operasi *Android*. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung *Google* atau dikenal sebagai *Open Handset Distribution* (OHD). (R & Yapie, 2013).

2.2.5.1. Arsitektur Android

Secara garis besar Arsitektur *Android* dapat dijelaskan sebagai berikut (Anwar et al., 2013):

1. *Application* dan *Widgets*

Application dan *Widgets* ini adalah layer dimana kita berhubungan dengan aplikasi saja, dimana biasanya kita *download* aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut. Di layer terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta,

browser, kontak, dan lain-lain. Hampir semua aplikasi ditulis menggunakan bahasa pemrograman Java.

2. *Application Frameworks*

Android adalah “*Open Development Platform*” yaitu *Android* menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi *resource*, menjalankan *service background*, mengatur alarm, dan menambah status *notifications*, dan sebagainya. Pengembang memiliki akses penuh menuju *API framework* seperti yang dilakukan oleh aplikasi kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (*reuse*). Sehingga bisa kita simpulkan *Application Frameworks* ini adalah layer dimana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi *Android*, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti *content providers* yang berupa sms dan panggilan telepon.

Komponen-komponen yang termasuk di dalam *Application Frameworks* adalah sebagai berikut: (Nazruddin, 2011).

- a. *Views*
- b. *Content Provider*
- c. *Resource Manager*
- d. *Notification Manager*

e. *Activity Manager*

3. *Libraries*

Libraries ini adalah layer dimana fitur-fitur *Android* berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya. Berjalan di atas *Kernel*, layer ini meliputi berbagai *library C/C++* inti seperti *Libc SSL*, serta: (Nazruddin, 2011).

- a. *Libraries* media untuk pemutaran media audio dan video
- b. *Libraries* untuk manajemen tampilan
- c. *Libraries Graphics* mencakup *SGL* dan *OpenGL* untuk grafis 2D dan 3D .
- d. *Libraries SQLite* untuk dukungan *database*.
- e. *Libraries SSL* dan *WebKit* terintegrasi dengan *web browser* dan *security*.
- f. *Libraries LiveWebcore* mencakup *modern web browser* dengan *engine embedded web view*
- g. *Libraries 3D* yang mencakup implementasi *OpenGL ES1.0 API's*.

4. *Android Run Time*

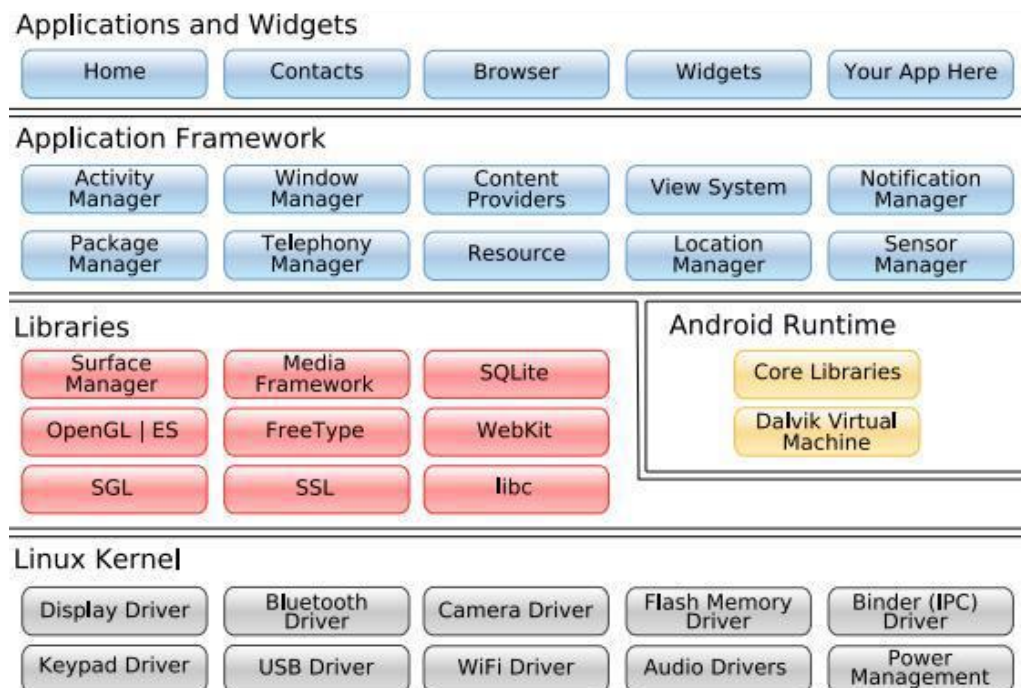
Layer yang membuat aplikasi *Android* dapat dijalankan dimana dalam prosesnya menggunakan Implementasi *Linux*. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi *Android*. Di dalam *Android Run Time* dibagi menjadi dua bagian yaitu (Nazruddin, 2011) :

- a. *Core Libraries*: Aplikasi *Android* dibangun dalam bahasa *Java*, sementara Dalvik sebagai virtual mesinnya bukan *Virtual Machine Java*, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menterjemahkan bahasa *Java/C* yang ditangani oleh *Core Libraries*.
- b. *Dalvik Virtual Machine*: Virtual mesin berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat *Linux Kernel* untuk melakukan *threading* dan manajemen tingkat rendah.

5. *Linux Kernel*

Linux Kernel adalah layer dimana inti dari sistem operasi *Android* itu berada. Berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem-sistem operasi *Android* lainnya. *Linux Kernel* yang digunakan *Android* adalah *Linux Kernel* release 2.6.

Arsitektur *Android* dapat digambarkan seperti pada Gambar 2.4 (Anwar et al., 2013):



Gambar 2.4. Arsitektur Android

Sumber : (Anwar et al., 2013)

2.2.6. *Android Studio*

Menurut Hasanah, Safriadi, & Tursina (2015), *Android Studio* adalah sebuah IDE yang bisa digunakan untuk pengembangan aplikasi *Android*, dan dikembangkan oleh *Google*. *Android Studio* merupakan pengembangan dari *Eclipse IDE*, dan dibuat berdasarkan *IDE Java* populer, yaitu *IntelliJ IDEA*. *Android Studio* direncanakan untuk menggantikan *Eclipse* ke depannya sebagai IDE resmi untuk pengembangan aplikasi *Android*.

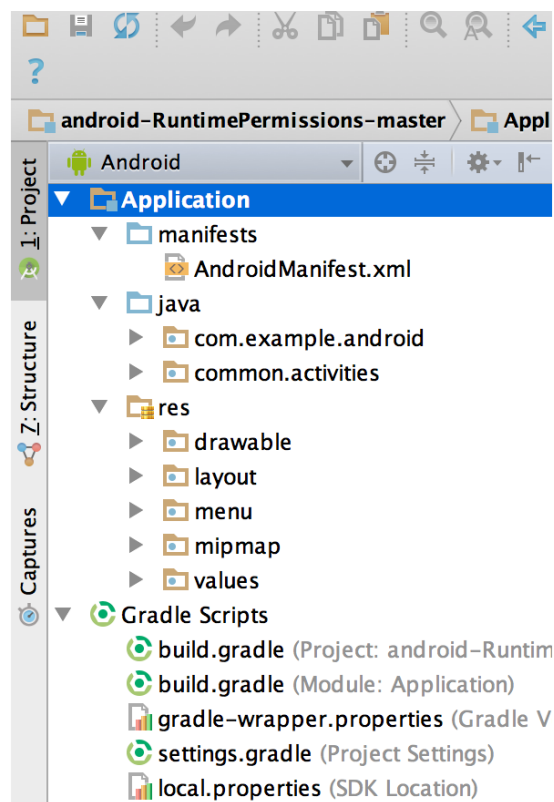
Sebagai pengembangan dari *Eclipse*, *Android Studio* mempunyai banyak fitur-fitur baru dibandingkan dengan *Eclipse IDE*. Berbeda dengan *Eclipse* yang menggunakan *ADT*, *Android Studio* menggunakan *Gradle* sebagai *build*

environment. Fitur-fitur lainnya adalah sebagai berikut (Hasanah, Safriadi, & Tursina, 2015) ;

1. Menggunakan *Gradle-based* build system yang fleksibel.
2. Bisa mem-build *multiple APK* .
3. Template support untuk *Google Services* dan berbagai macam tipe perangkat.
4. *Layout editor* yang lebih bagus.

Berikut adalah beberapa fitur yang terdapat pada aplikasi *Android Studio* (Android Developer, 2018):

2.2.6.1. Struktur Proyek *Android Studio*



Gambar 2.5. File proyek di tampilan Android.

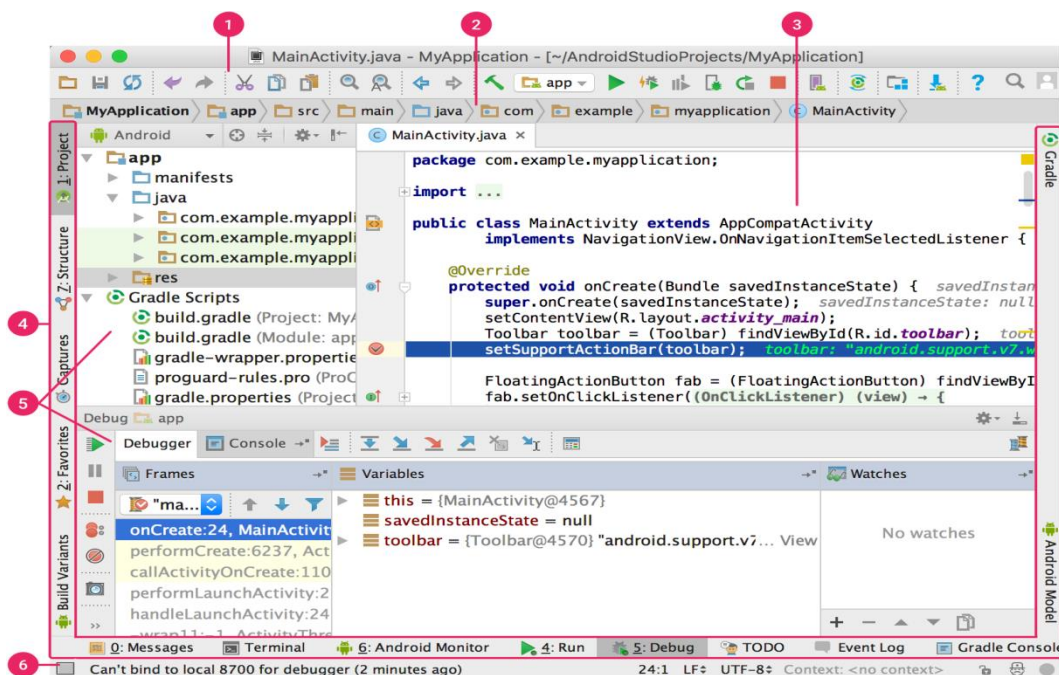
Sumber : (Android Developer, 2018)

Semua file versi terlihat di bagian atas di bawah **Gradle Scripts** dan masing-masing modul aplikasi berisi folder berikut:

- a. **manifests**: Berisi file *AndroidManifest.xml*.
- b. **java**: Berisi file kode sumber Java, termasuk kode pengujian JUnit.
- c. **res**: Berisi semua sumber daya bukan kode, seperti tata letak XML, string UI, dan gambar bitmap.

2.2.6.2. Antar Muka Pengguna *Android Studio*

Jendela utama *Android Studio* terdiri dari beberapa bidang logika yang diidentifikasi dalam gambar 2.6 :



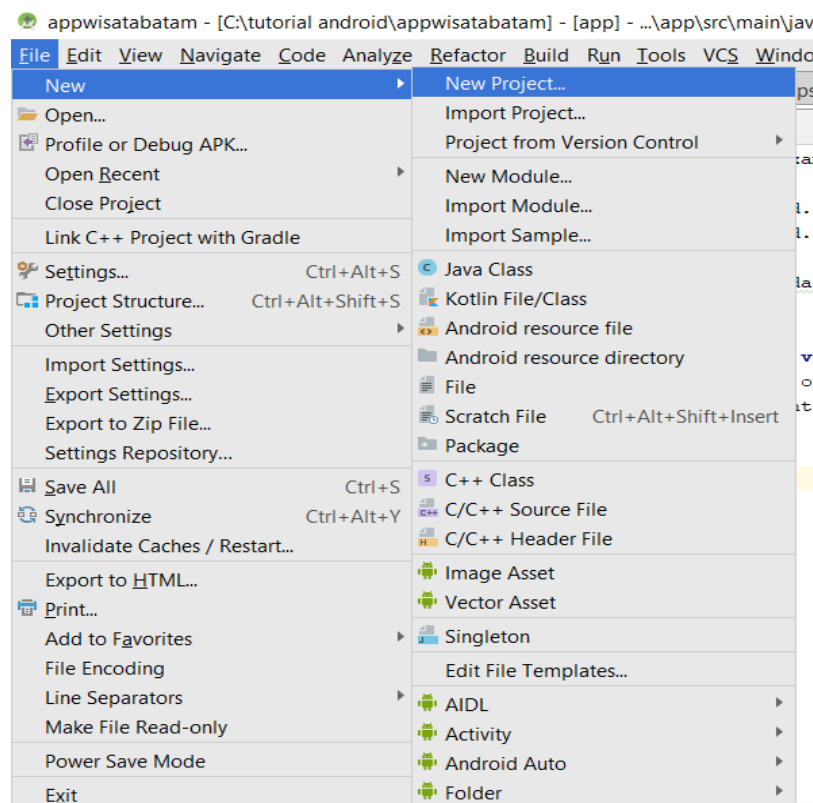
Gambar 2.6. Jendela utama *Android Studio*.

Sumber : (Android Developer, 2018)

- a. **The toolbar** memungkinkan Anda untuk melakukan berbagai jenis tindakan, termasuk menjalankan aplikasi dan meluncurkan alat Android.
- b. **Navigation bar** membantu Anda bernavigasi di antara proyek dan membuka file untuk diedit. Bilah ini memberikan tampilan struktur yang terlihat lebih ringkas dalam jendela **Project**.
- c. **Editor window** adalah tempat Anda membuat dan memodifikasi kode. Bergantung pada jenis file saat ini, editor dapat berubah. Misalnya, ketika melihat file tata letak, editor menampilkan *Layout Editor*.
- d. **Bar tools window** muncul di luar jendela IDE dan berisi tombol yang memungkinkan Anda meluaskan atau menciutkan jendela alat individual..
- e. **Window tool** memberi Anda akses ke tugas tertentu seperti pengelolaan proyek, penelusuran, kontrol versi, dan banyak lagi. Anda bisa meluaskan dan juga menciutkannya.
- f. **The status bar** menampilkan status proyek Anda dan IDE itu sendiri, serta setiap peringatan atau pesan.

2.2.6.3. Membuat Proyek Baru *Android Studio*

Membuat proyek baru dilakukan untuk mengarahkan lalu menyatukan dan menyimpan seluruh elemen program atau proyek yang akan anda buat pada satu kesatuan didalam sebuah *folder* pada *workspace* yang telah di tentukan.



Gambar 2.7. Membuat Project Baru

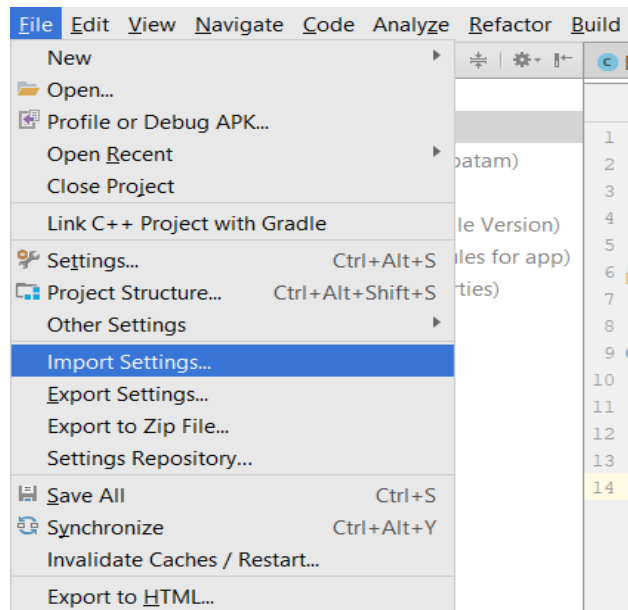
Sumber : (Android Developer, 2018)

Membuat proyek baru dapat dilakukan dengan cara memilih *New > new project* (gambar 2.7.) ketika anda telah membuka *android studio* lalu memilih *workspace* sesuai dengan keinginan.

2.2.6.4. Import Projek Android Studio

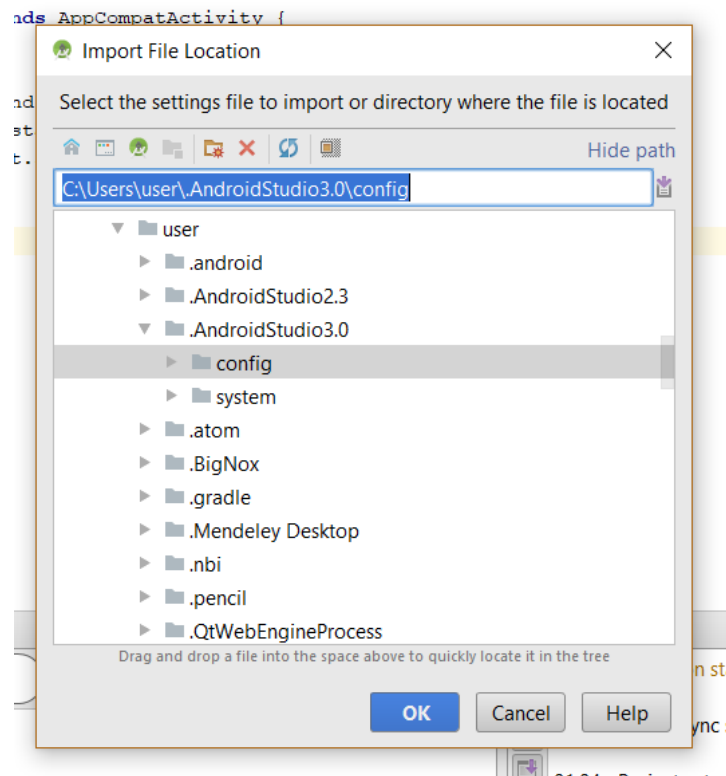
Java dibuat dengan motto “*write once, run anywhere*” yang artinya java dapat berdiri sendiri dan dapat beradaptasi dengan lingkungan *Operating system* (OS). Dengan demikian java mempermudah anda dalam pembuatan program, proses running, maupun pengaplikasian program.

Import proyek adalah salah satu fasilitas yang memwadahi hal tersebut, anda dapat membaca dan menjalankan proyek yang dibuat dengan *Operating System* (OS) yang berbeda dengan yang digunakan. Berikut adalah langkah-langkah yang digunakan dalam melakukan *import project*:



Gambar 2.8. Import Project

Sumber : Hasil Penelitian



Gambar 2.9. Import project tahap 2

Sumber : Hasil Penelitian

- a. *Copy folder project* yang akan diimport, lalu paste pada *project android studio* pada komputer anda.
- b. Buka aplikasi *android studio* anda lalu klik *File > Import settings* seperti gambar 2.8 di atas
- c. Ketika anda memilih *import* maka akan muncul pilihan *import, drop down* pilihan *general* seperti gambar 2.9. diatas Untuk memilih *project*.

2.2.7. *SQLite*

Menurut Fatansyah (2012), *Database* tabel-tabel yang terdiri dari *field-field* atau sebagai tempat penyimpanan data dan informasi oleh beberapa unit organisasi, dimana *database* mempunyai kecenderungan berkembang sejalan dengan perkembangan organisasi, sehingga interaksi antar unit akan bertambah besar yang menyebabkan ini formasi yang dibutuhkan juga akan semakin bertambah. (Kevin et al., 2016)

Menurut Mufti, (2015), *SQLite* merupakan sebuah software manajemen *database* yang simple dalam penggunaanya dan dapat disisipkan dalam aplikasi dalam *database Android*. Apabila kita ingin mengakses database di *web server*, kita perlu koneksi internet, tetapi pada *SQLite* kita tidak perlu tersambung dengan internet. Kita dapat menambahkan, mengedit, dan menghapus data di dalamnya, serta dapat menjalankan sebuah syntak *SQL*.

2.2.8. *Blackbox Testing*

Menurut Utama et al., (2016), *Blackbox testing* merupakan pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian *blackbox* dilakukan dengan membuat kasus uji yang

bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sudah sesuai dengan spesifikasi yang dibutuhkan.

Metode ujicoba *blackbox* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya (Djafar, 2015):

1. Fungsi-fungsi yang salah atau hilang.
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses *database* eksternal
4. Kesalahan performa
5. kesalahan inisialisasi dan terminasi

2.3. Penelitian Terdahulu

Pada tinjauan penelitian terdahulu akan dibahas secara lengkap jurnal dan artikel yang mendukung sebagai dasar pembahasan interpretasi penelitian pada bahan sebelumnya.

Tabel 2.5. Penelitian Terdahulu

No	Nama Peneliti	Judul	Jurnal Penelitian	Metode Penelitian	Hasil Penelitian
1.	Kushwaha & Kushwaha , 2011	<i>Location Based Services using Android Mobile Operating System</i>	<i>International Journal of Advances in Engineering & Technology</i> , ISSN: 2231-1963	-	Hasil penelitian Aplikasi LBS dapat membantu pengguna menemukan rumah sakit, sekolah, pompa bensin atau fasilitas lainnya ditunjukkan kepada pengguna. Sama seperti perangkat GPS lokasi diperbarui posisinya.
2.	Nugroho & Supriyanto , 2012	Penentuan Rute Terdekat dan Lokasi Rumah Sakit di Jawa Tengah Menggunakan <i>Location Based Service</i> (LBS) pada Platform <i>Android</i>	Jurnal Dinamika Informatika, Vol.4 No. 1, ISSN 2085-3343	Tahap pengembangan sistem ini dipakailah suatu metode <i>prototype</i> , yang terdiri dari tahap analisis, mendesain <i>prototype</i> , evaluasi dan penerapan <i>prototype</i> .	Hasil penelitian aplikasi ini, para peserta asuransi Askes dan Jamsostek dapat mengetahui letak rumah sakit asuransi terdekat disekitarnya beserta informasi pendukung hanya bermodalkan ponsel Android.
3.	Singhal & Shukla, 2012	<i>Implementation of Location based Services in Android using GPS and Web Services</i>	<i>International Journal of Computer Science Issues</i> , Vol. 9, Issue 1, No 2, ISSN 1694 - 0814	Metode Penelitian ini yaitu implementasi <i>Location based services</i> melalui <i>Google Web Services</i> dan <i>Walk Score Transit API</i> di handphone <i>Android</i> .	Hasil penelitian ini <i>Location based Services</i> menawarkan banyak keuntungan bagi ponsel untuk mengambil informasi tentang lokasi mereka saat ini dan memproses data tersebut untuk mendapatkan lebih banyak informasi bermanfaat di dekat mereka lokasi.

No	Nama Peneliti	Judul	Jurnal Penelitian	Metode Penelitian	Hasil Penelitian
4	Hidayat & Februariyanti, 2013	Aplikasi <i>Location Based Service</i> (LBS) Pencarian Lokasi Taxi pada <i>Android</i> di Kota Semarang	Jurnal Dinamika Informatika , Vol.5 No. 1, ISSN 2085-3343	Tahap pengembangan sistem ini dipakailah suatu metode <i>prototype</i> . yang terdiri dari tahap design, tahap pembuatan program, tahap evaluasi, dan hasil	Hasil penelitian aplikasi ini, pengguna dapat dengan mudah memesan taxi dan melihat lokasi basecamp taxi terdekat, serta memiliki informasi langsung mengenai nomor telepon perusahaan taxi dan dapat langsung menelepon perusahaan
5.	Kasuma, Mulyani & Yapie, 2013	Aplikasi <i>Location Based Service</i> (LBS) Taman Mini Indonesia Indah (TMII) Berbasis <i>Android</i>	Yogyakarta, , ISSN: 1907 - 5022	Metode penelitian yang digunakan yaitu pendekatan SDLC, yang terdiri dari fase identifikasi, fase analisis, fase perancangan, dan fase uji.	Hasil Penelitian Melalui Aplikasi Peta Wisata TMII <i>Android</i> ini, pengunjung akan lebih mudah dan tepat dalam menemukan lokasi objek wisata yang tersebar di wilayah TMII.
6.	Riza, Ikhwana & Retnadi, 2013	Pengembangan Aplikasi Pencarian Lokasi Objek Wisata Terdekat di Kabupaten Garut Berbasis <i>Android</i> .	Jurnal Algoritma Sekolah Tinggi Teknologi Garut, ISSN : 2302-7339 Vol.10 No. 1	Metodologi yang di pakai dalam perancangan ini adalah dengan Object Oriented Desain (OOD) dari Unified Approach (UA) Ali Bahrami	Dari hasil penelitian Aplikasi pencarian Lokasi Objek Wisata Terdekat di Kabupaten Garut Berbasis <i>Android</i> ini memudahkan pengguna dalam memperoleh informasi mengenai lokasi wisata disertai dengan fasilitas-fasilitas yang berada di sekitar objek wisata yang dituju.

No	Nama Peneliti	Judul	Jurnal Penelitian	Metode Penelitian	Hasil Penelitian
7.	Anwar, Nugroho, & Lestariningsih, 2013	Perancangan dan Implementasi Aplikasi Mobile Semarang Guidance pada Android.	Jurnal Dinamika Informatika – Vol.5 No. 2, ISSN 2085-3343	Penelitian ini menggunakan model pengembangan SDLC. Model analisisnya yang menggunakan Use Case, Activity, Class Diagram, E-R Diagram dan kamus data.	Hasil penelitian aplikasi ini, dapat memberikan kemudahan dalam mengetahui letak dan posisi geografis tempat wisata terdekat disekitar pengguna beserta informasi pendukung melalui ponsel Android.
8.	Razaq & Arief, 2014	Sistem Informasi Publik Layanan Kesehatan menggunakan Metode Location Based Service di Kota Semarang	Jurnal Teknologi Informasi Dinamik Vol 19, No.1, 59-67 ISSN : 0854-9524	Penelitian ini menggunakan model siklus hidup (SDLC). Model analisis sistem menggunakan Use Case, Activity diagram, Class diagram, Kamus data & ERD	Hasil penelitian aplikasi ini, dapat memberikan kemudahan, kecepatan, dan ketepatan dalam mengetahui posisi geografis lokasi layanan kesehatan terdekat di sekitar pengguna beserta informasi melalui perangkat mobile Android.
9.	Mudzakir & Arifudin, 2015	Aplikasi Location Based Service Fasilitas Umum Berbasis Android	UNNES Journal of Mathematics , ISSN 2252-6943	Metode yang digunakan untuk pengembangan aplikasi ini yaitu menggunakan model prototype yang terdiri dari fase identifikasi, fase analisis, fase perancangan, dan fase uji.	Hasil penelitian Aplikasi ini mampu untuk memberikan informasi terkait suatu lokasi layanan kesehatan serta beberapa fitur yang memanfaatkan <i>location based service</i> .
10.	Sukerta, Linawati & Wirastuti, 2015	Sistem Aplikasi Location Based Service untuk Pengembangan Kota Cerdas.	Jurnal Teknologi Elektro, Vol.14, No.1, ISSN 1693-2951	Aplikasi ini dibangun dengan arsitektur Client - Server berbasis web. Pada sisi client menggunakan antarmuka web maupun mobile dengan sistem operasi <i>Android</i> .	Hasil Penelitian aplikasi Informasi lokasi permasalahan di lapangan dapat segera diketahui secara tepat berdasarkan koordinatnya, sehingga penanganannya dapat dilakukan dengan segera.

No	Nama Peneliti	Judul	Jurnal Penelitian	Metode Penelitian	Hasil Penelitian
11.	Nama et al., 2015	<i>Design and Implementation Web Based Geographic Information System for Public Services in Bandar Lampung City</i>	<i>International Conference on Science in Information Technology</i> , IEEE 978-1-4799-8386-5/15	Metode Penelitian yang dilakukan yaitu, Analisis Kebutuhan, UML Model Design dan Implementasi WEB GIS Application	Hasil penelitian Aplikasi ini bisa dengan mudah diintegrasikan dengan portal web pemerintah kota yang ada untuk meningkatkan pelayanan publik yang lebih baik bagi warga.
12.	Utama et al., 2016	Pembuatan Aplikasi Memantau Lokasi Anak Berbasis Android Menggunakan Location Based Service	Jurnal Teknologi & Sistem Komputer, Vol.4, No.1, (e-ISSN: 2338-0403)	Metode penelitian yang digunakan adalah SDLC (<i>Software Development Life Cycle</i>) dengan teknik <i>waterfall</i> .	Hasil penelitian ini dapat menampilkan lokasi pengguna dan juga anggota keluarga terutama anak, sehingga orang tua yang menggunakan aplikasi ini dapat melihat lokasi anak mereka saat ini.

Sumber : Hasil Penelitian