BAB 2
LANDASAN TEORI

2.1. Pengertian Tanaman Jeruk

Jeruk merupakan buah-buahan yang mempunyai nilai ekonomi penting dan nilai kesehatan yang berarti karena mengandung nilai gizi yang sangat tinggi vitamin A dan vitamin C . Buah jeruk dapat dikonsumsi langsung sebagai buah segar atau jus dan dapat pula diolah menjadi sirup. Produktivitas jeruk rata-rata di Indonesia masih rendah, Lagi pula ada indikasi bahwa setelah berumur 7 tahun produktivitas jeruk cenderung menurun. Kemunduran produktivitas diduga karena kekurangan air, gangguan, perakaran karena kondisi tanah, hama, dan penyakit, dan lain-lain. [1].

2.1.1 Bagian Pohon Jeruk

1. Akar

Tanaman jeruk manis mempunyai akar tunggang panjang dan akar serabut (bercabang pendek kecil) serta akar – akar rambut. Bila akar tunggang mencapai tanah yang keras atau tanah yang terendam air, maka pertumbuhannya akan berhenti. Tetapi bila tanahnya gembur, panjang akar tunggang bisa mencapai 4 meter. Akar cabang yang mendatar bisa mencapai 6 – 7 meter. Perakaran jeruk manis tergantung pada banyaknya unsur hara di dalam tanah dan umumnya di kedalaman 0,15 – 0,50 meter.

2. Pohon

Pohon jeruk manis berbentuk bulat dan tingginya mencapai 5 – 15 meter.

3. Daun

Daun jeruk manis berwarna hijau tua dan tidak merangas. Berbentuk bulat telur (elips) dengan panjang ± 5 – 15 cm, ujungnya runcing sedikit tumpul dan sedikit berlekuk. Tepi daun bergirigi halus dan pada kedua permukaannya tidak berbulu.

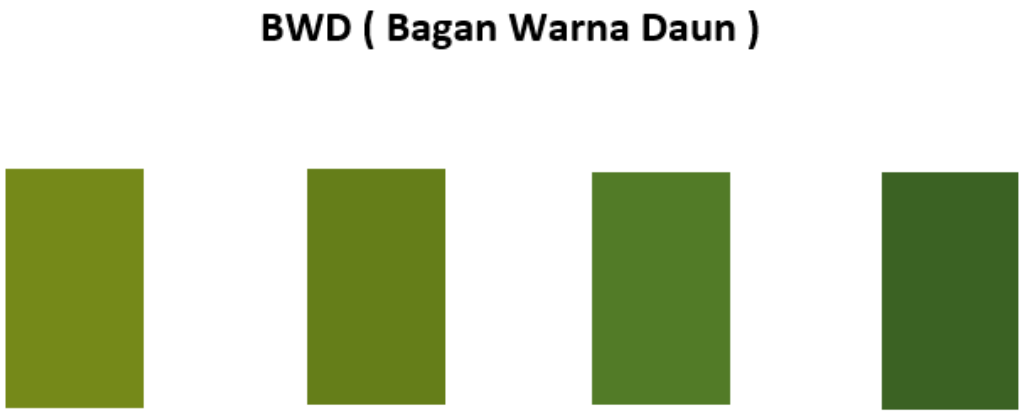
4. Bunga

Bunga jeruk manis adalah bunga majemuk seperti anak payung, tandan atau malai kebanyakan berkelamin 2, kelopak dan mahkota bunga berjumlah 4 – 5, dan berdaun lepas. Bunga jeruk manis berwarna putih, keluar dari ketiak daun atau pucuk ranting yang masih muda, berbau harum dan banyak mengandung nectar atau madu.

5. Buah

Jeruk manis berkulit tebal ± 4 mm, bentuk buah bulat, warna kulit luar hijau sampai jingga atau orange, warna daging buah kuning pucat sampai dengan kuning segar.[1]

2.1.2. Bagan Warna Daun Jeruk



Pemupukan :

SKALA	PUPUK KANDANG	ZA	DS	ZK
1	20kg	2500gram	450gram	1000gram
2	14kg	2000gram	350gram	900gram
3	10kg	1500gram	250gram	750gram
4	8kg	1000gram	200gram	600gram

Gambar 2.1 Bagan Warna Daun

2.2. Aplikasi

Perangkat lunak aplikasi yaitu perangkat lunak yang digunakan untuk membantu pemakai komputer untuk melaksanakan pekerjaannya. Jika ingin mengembangkan program aplikasi sendiri, maka untuk menulis program aplikasi

tersebut, dibutuhkan suatu bahasa pemrograman yang dapat berbentuk assembler, compiler ataupun interpreter.

Bahasa pemrograman berfungsi untuk menulis program dengan bahasa yang lebih mudah, dan selanjutnya akan diterjemahkan ke dalam bahasa mesin supaya bisa dimengerti oleh komputer. Bila hendak mengembangkan suatu program aplikasi untuk memecahkan permasalahan yang besar dan rumit, maka supaya program aplikasi tersebut dapat berhasil dengan baik, maka dibutuhkan prosedur dan perencanaan yang baik dalam mengembangkannya. Saat ini terdapat program-program aplikasi yang tersedia dalam bentuk paket-paket program. Beberapa perusahaan perangkat lunak telah memproduksi paket-paket perangkat lunak yang mempunyai reputasi internasional. Program-program paket tersebut dapat diandalkan, dapat memenuhi kebutuhan pemakai, dirancang dengan baik, relatif bebas dari kesalahan-kesalahan, user friendly (mudah digunakan), mempunyai dokumentasi manual yang memadai, mampu dikembangkan untuk kebutuhan mendatang, dan didukung perkembangannya.

Dari pengertian diatas peneliti menyimpulkan bahwa sebuah aplikasi dapat dibuat dan dikembangkan dengan tujuan untuk melakukan tugas yang bersifat umum atau juga dapat dikembangkan untuk melakukan tugas yang bersifat spesifik/khusus. Hal ini didasarkan pada permasalahannya yang bisa saja ditemui bersifat unik sehingga tidak ada paket-paket program yang sesuai untuk digunakan. Oleh karena itu dengan terpaksa diperlukan pengembangan program aplikasi itu sendiri.[2]

2.3. *Smartphone*

Menurut Williams dan Sawyer smartphone adalah telepon selular dengan mikroprosesor, memori, layar dan modem bawaan. Smartphone merupakan ponsel multimedia yang menggabungkan fungsionalitas Personal Computer(PC) dan handset sehingga menghasilkan gadget yang mewah dimana terdapat pesan teks, kamera, pemutar musik, video, game, akses email, tv digital, search engine, pengelola informasi pribadi, fitur GPS, jasa telepon internet, dan bahkan terdapat telepon yang juga berfungsi sebagai kartu kredit.

Didalam penelitian ini smartphone digunakan untuk membantu aplikasi dalam mengakses perangkat keras melalui Application Programming Interface (API) yang terdapat pada sebuah sistem operasi Android untuk kemudian dapat memberikan data monitoring. *Monitoring* [2]

2.4. Android

Android adalah sebuah kumpulan perangkat lunak yang bersifat open source yang mencakup sistem operasi, middleware, dan aplikasi kunci bersama dengan satu set Application Programming Interface (API) untuk menulis aplikasi mobile yang dapat membentuk tampilan, rasa, dan fungsi dari sebuah perangkat mobile. Awalnya, Google Inc. membeli Android Inc yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/smartphone. Para pendiri Android Inc. bekerja pada Google untuk memulai membangun platform Android secara lebih intensif, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Pada tanggal 12 November 2007 Google bersama Open Handset Aliance (OHA) merilis Google Android Software Development Kit (SDK). Paket SDK yang dirilis untuk mengembangkan aplikasi yaitu sistem operasi, middleware, dan aplikasi utama untuk perangkat mobile. Dengan dirilisnya SDK tersebut membuka kesempatan kepada pengembang untuk mengembangkan sebuah aplikasi berbasis Android.

Hingga awal tahun 2017 terdapat beberapa versi Android yang tersedia untuk digunakan, diantaranya adalah sebagai berikut:

1. Android versi 1.1
2. Android versi 1.5 Cupcake
3. Android versi 1.6 Donut
4. Android versi 2.0/2.1 Éclair
5. Android versi 2.2 Froyo
6. Android versi 2.3 GingerBread
7. Android versi 3.0/3.1/3.2 Honeycomb
8. Android versi 4.0 Ice Cream Sandwich
9. Android versi 4.1.2/4.2.2/4.3.1 Jelly Bean
10. Android versi 4.4.2 Kitkat
11. Android versi 5.0.1/5.1.1 Lollipop

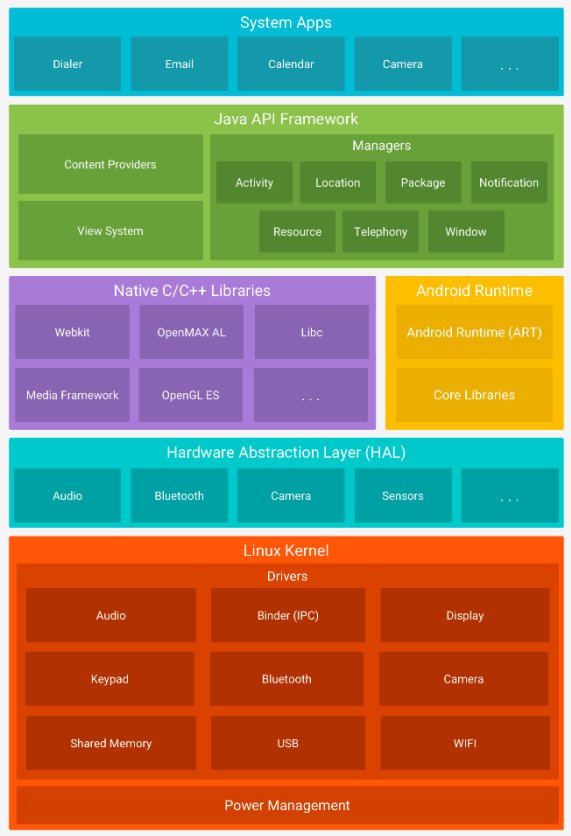
12. Android versi 6.0 Marshmallow

13. Android versi 7.0/7.1.1.1 Nougat

Didalam penelitian ini peneliti memilih versi sistem operasi Android Lollipop (API Level 22) ke atas karena pada sistem operasi tersebut telah terdapat interface untuk membaca ID unik pada banyak slot kartu SIM. [3]

2.5. Arsitektur Android

Secara garis besar komponen utama yang terdapat pada arsitektur Android dapat dilihat pada Gambar 2.2.



Sumber Gambar : https://developer.android.com/guide/platform/index.html

Gambar 2.2 Arsitektur Android

1. Linux Kernel Linux

kernel adalah layer dimana inti dari sistem operasi Android berada. Pada linux kernel terdapat file-file sistem yang mengatur sistem processing, memory, resource, drivers, dan sistem-sistem operasi Android lainnya. Dengan

menggunakan linux kernel akan membolehkan Android untuk memperoleh keuntungan fitur kunci keamanan dan membolehkan manufaktur perangkat untuk mengembangkan driver perangkat keras untuk kernel umum.

2. Hardware Abstraction Layer (HAL)

Hardware Abstraction Layer menyediakan interface standard yang mendukung kapabilitas hardware sebuah perangkat kedalam tingkatan framework API Java yang lebih tinggi. HAL berisi beberapa modul library yang mengimplementasikan interface masing-masing untuk jenis komponen hardware khusus seperti modul kamera atau Bluetooth. Ketika framework API membuat pangilan untuk mengakses hardware, sistem Android memuat modul library untuk komponen hardware tersebut.

3. Android Runtime

Android Runtime adalah layer yang membuat aplikasi Android dapat dijalankan dimana dalam prosesnya menggunakan implementasi Linux. Untuk perangkat yang berjalan pada versi Android 5.0 (API 21) atau keatas, setiap aplikasi berjalan didalam proses dan instansi Android Runtime (ART) masing-masing. ART ditulis untuk menjalankan beberapa virtual machine pada perangkat memori rendah dengan mengeksekusi file DEX. Berdasarkan Android versi 5.0 (API 21), Dalvik dulunya adalah Android Runtime. Apabila aplikasi berjalan baik pada ART, maka seharusnya berjalan baik pula pada Dalvik.

4. Native C/C++ Libraries

Beberapa komponen dan service dari sistem android seperti ART dan HAL dibuat dengan kode native yang membutuhkan native library yang ditulis dalam bahasa C dan C++. Platform Android menyediakan framework API Java untuk memaparkan fungsionalitas dari native library tersebut terhadap aplikasi.

5. Java API Framework

Seluruh set fitur pada sistem operasi Android tersedia untuk para pengembang melalui API yang ditulis dalam bahasa Java. Dalam hal ini, pengembang memiliki akses penuh terhadap framework API yang sama yang digunakan oleh aplikasi pada sistem Android. API-API tersebut menyediakan kebutuhan dalam membangun sebuah aplikasi yang mencakup :

1. Sistem View yang kaya dan extensible yang dapat digunakan untuk membuat UI aplikasi, seperti list, grid, text box, button, dan sebagainya.
2. Resource Manager yang menyediakan akses terhadap sumberdaya noncode seperti localized string, graphic, dan file layout.
3. Notification Manager yang membolehkan aplikasi untuk menampilkan kustom alert didalam status bar.
4. Activity Manager yang mengatur lifecycle sebuah aplikasi dan menyediakan navigasi back stack umum.
5. Content Provider yang membolehkan aplikasi untuk mengakses data dari aplikasi lain seperti aplikasi kontak atau untuk membagikan data aplikasi itu sendiri.

6. System Apps

Android dibuat dengan satu set aplikasi bawaan seperti email, SMS messaging, calendar, internet browsing, contacts, dan lain lain. Aplikasi yang disertakan oleh platform tidak memiliki status spesial. Aplikasi fihak ketiga yang diinstal oleh pengguna dapat saja menjadi aplikasi default pengguna. Fungsi dari system apps adalah bertindak sebagai aplikasi untuk pengguna dan juga penyedia kapabilitas kunci untuk pengembang yang ingin mengakses melalui aplikasi miliknya sendiri.

Berdasarkan komponen arsitektur Android diatas diketahui bahwa sistem aplikasi yang dikembangkan oleh pengembang aplikasi dapat mengakses API dari lapisan framework Java Android. Dengan demikian aplikasi yang dikembangkan dapat digunakan mengakses hardware pada perangkat Android.

2.6. Komponen Aplikasi Android

Aplikasi Android ditulis dalam bahasa pemrograman Java. Kode Java dikompilasi bersama dengan data file yang dibutuhkan oleh aplikasi dimana prosesnya dikemas oleh tools yang dinamakan “apt tools” kedalam paket Android sehingga menghasilkan file dengan ekstensi apk. File apk itulah yang kita sebut dengan aplikasi dan nantinya dapat dipasang pada perangkat mobile.

Terdapat empat jenis komponen pada aplikasi Android yaitu:

1. Activity

Activity akan menyajikan User Interface (UI) kepada pengguna sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi Android bisa jadi hanya memiliki satu activity, tetapi umumnya aplikasi memiliki banyak activity yang bergantung pada tujuan aplikasi dan desain aplikasi itu sendiri. Untuk berpindah dari satu activity ke activity lain dapat dilakukan menggunakan sebuah trigger seperti klik tombol pada tampilan aplikasi.

2. Service

Service tidak memiliki Graphic User Interface (GUI), tetapi service berjalan secara background. Sebagai contoh dalam memutar musik, service mungkin memutar musik atau mengambil data dari jaringan, tetapi setiap service harus berada dalam kelas induknya. Apabila sebuah pemutar musik sedang memutar lagu dari list yang ada, aplikasi akan memiliki dua atau lebih activity yang memungkinkan pengguna untuk memutar sambil memilih lagu baru. Untuk menjaga musik tetap berjalan sebuah activity dapat menjalankan service.

3. Broadcast Receiver

Broadcast receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh broadcast seperti notifikasi zona waktu telah berubah, baterai lemah, gambar berhasil diambil oleh kamera, dan lain-lain. Aplikasi juga dapat menginisialiasi broadcast misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah diunduh ke perangkat dan siap untuk digunakan.

4. Content Provider

Content provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti database

SQLite. *Content provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *activity.*

Didalam penelitian ini menggunakan seluruh komponen aplikasi diatas. Activity digunakan peneliti agar dapat memberikan Antar Muka bagi pengguna aplikasi, service digunakan untuk proses yang bergerak tanpa Antar Muka, broadcast receiver digunakan untuk membantu mendeteksi SMS masuk atau terjadi pergantian kartu SIM, dan content provider digunakan untuk membantu melakukan query terhadap data seperti kontak pada kartu SIM. [3]

2.7. Android SDK (Software Development Kit)

Android SDK (Software Development Kit) adalah tools API (Application Programming Interface) yang diperlukan untuk mulai mengembangkan aplikasi pada platform Android. Android SDK disediakan sebagai alat bantu untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. Sebagai platform aplikasi-netral, Android memberi kesempatan untuk membuat aplikasi yang dibutuhkan.

Android SDK versi resmi dapat ditemukan di situs resmi Google. Namun terdapat juga SDK buatan pihak ketiga seperti App Inventor untuk Android, Basic4android, Corona SDK, RubyMotion, dan Xamarin. SDK tersebut dapat digunakan untuk menulis program Android, namun tentunya bahasa pemrograman yang digunakan menyesuaikan dengan SDK masing-masing.

Setiap kali Google merilis versi baru dari Android maka sebuah SDK yang sesuai juga dirilis. Untuk dapat menulis program dengan fitur terbaru maka pengembang aplikasi Android harus mengunduh dan menginstal Android SDK sesuai versi masing-masing untuk smartphone tertentu. Didalam penelitian ini penulis menggunakan Android SDK resmi dari Google dengan versi minimum 25. [4]

2.8. Android Studio

Android Studio adalah Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. Selain merupakan

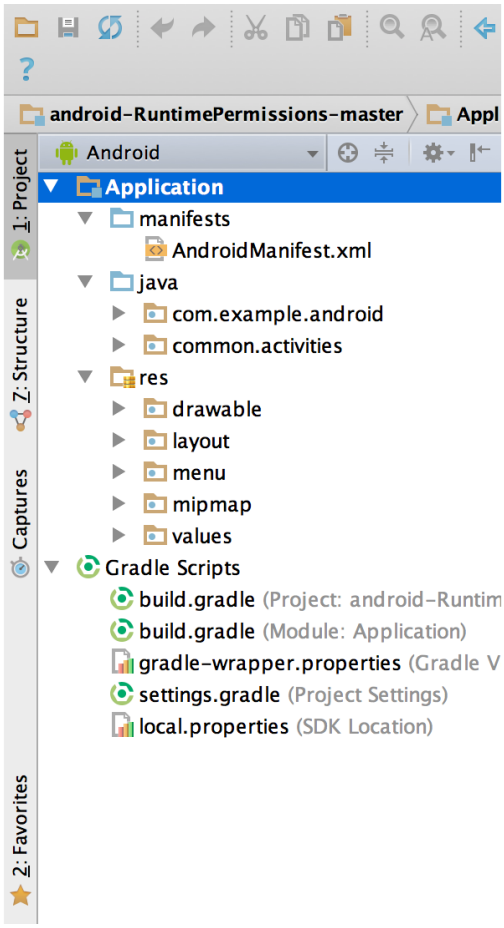
editor kode IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas saat membuat aplikasi Android, misalnya:

1. Sistem pembuatan berbasis Gradle yang fleksibel
2. Emulator yang cepat dan kaya fitur
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android
4. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
5. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh
6. Alat penguji dan kerangka kerja yang ekstensif
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine

Setiap proyek di Android Studio berisi satu atau beberapa modul dengan file kode sumber dan file sumber daya. Jenis-jenis modul mencakup:

1. Modul aplikasi Android
2. Modul perpustakaan
3. Modul Google App Engine

Secara default, Android Studio menampilkan file proyek Anda dalam tampilan proyek Android seperti yang ditunjukkan dalam gambar 2.2. Tampilan ini diatur menurut modul untuk memberi akses cepat ke file sumber kunci proyek Anda.



Sumber Gambar : https://developer.android.com/studio/intro/index.html

Gambar 2.3 File Proyek di Tampilan Android

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

1. manifests yang berisi file AndroidManifest.xml.
2. java yang berisi file kode sumber Java, termasuk kode pengujian JUnit.
3. res: Berisi semua sumber daya bukan kode, seperti tata letak XML, string UI, dan gambar bitmap.

Struktur proyek Android pada disk berbeda dari representasi rata ini. Untuk melihat struktur file sebenarnya dari proyek ini, pilih Project dari menu tarik turun Project (dalam gambar 2.2, struktur ditampilkan sebagai Android).

Android Studio digunakan peneliti dibandingkan IDE lain karena Android Studio telah memiliki dukungan penuh dari Google. Hal tersebut memudahkan pengembang dalam melakukan integrasi seperti penggunaan sebuah library yang melalui file gradle cukup dengan mendaftarkan satu baris program saja. Adapun versi Android studio yang digunakan peneliti yaitu versi 2.3.2. [5]

2.9. Bahasa Pemograman Java

Java telah berkembang dari semula ditujukan untuk pemrograman applet di web browser menjadi bahasa pemrograman pengembangan aneka ragam aplikasi, mulai dari yang berjalan di handheld devices seperti handphone, PDA, sampai aplikasi tersebar skala enterprise di beragam komputer server. Java merupakan bahasa orientasi objek untuk pengembangan aplikasi mandiri, aplikasi berbasis internet, aplikasi untuk perangkat cerdas yang dapat berkomunikasi lewat internet/jaringan komunikasi. Java tidak lagi hanya bahasa untuk membuat applet yang memperindah halaman web tapi Java telah menjadi bahasa untuk pengembangan aplikasi skala enterprise berbasis jaringan besar.

Menurut Budi Raharjo, Imam Heryanto, Arif haryono java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis Java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (general purpose), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi Java mampu berjalan di beberapa platformsistem operasi yang berbeda, Java dikenal pula dengan slogannya, "Tulis sekali, jalankan di mana pun". Saat ini Java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web. [6]

Java bukan merupakan satu-satunya bahasa pemrograman yang dapat digunakan untuk mengembangkan aplikasi berbasis Android. Adapun bahasa lain seperti Xamarin yang menggunakan bahasa pemgrograman C# yang dapat digunakan sebagai opsi alternatif dalam membangun aplikasi berbasis Android. Namun disini peneliti menggunakan bahasa pemrograman Java karena lebih mudah dalam mencari contoh dokumentasi dan merupakan bahasa yang didukung oleh IDE Android Studio resmi dari Google.[6]

2.10. Web Service

Web service merupakan kumpulan logika aplikasi berisikan objek dan metode yang terletak didalam web server yang terhubung ke internet sehingga dapat diakses menggunakan protokol HTTP dan SOAP (Simple Object Access Protocol). Web service dapat digunakan untuk menyelesaikan masalah seperti aplikasi baru yang ingin memeriksa data transaksi keuangan dari sebuah database yang dikelola oleh aplikasi yang dibangun dengan bahasa pemrograman berbeda. Tujuan dari teknologi web service ini adalah untuk memudahkan beberapa aplikasi atau komponen aplikasi untuk saling berhubungan dengan aplikasi lain dalam sebuah organisasi dengan menggunakan standar yang tidak terikat platform dan tidak terikat akan bahasa pemrograman yang digunakan.

SOAP merupakan protokol standar yang ditujukan untuk pertukaran informasi. Protokol ini melakukan pemanggilan metode dalam bahasa XML (Extensible Markup Language). Selain XML, juga terdapat opsi lain yaitu menggunakan format pertukaran data JSON (JavaScript Object Notation). Dengan demikian SOAP adalah suatu mekanisme sederhana untuk melakukan pertukaran struktur dan tipe informasi dalam lingkungan yang tersebar dan terdistribusi menggunakan XML/JSON. Dokumen SOAP yang digunakan untuk melakukan request disebut SOAP request sedangkan dokumen SOAP yang diperoleh dari web service disebut dengan SOAP response.

Untuk menjalankan fungsinya, web service memerlukan agen. Agen adalah bagian perangkat lunak atau perangkat keras yang mengirimkan dan menerima pesan. Agen dapat ditulis dengan berbagai bahasa pemrograman. Dan dapat berganti-ganti bahasa pemrograman dengan fungsi yang sama. Tujuan web service

adalah untuk menyediakan beberapa fungsi atas nama pemiliknya, seseorang atau organisasi seperti bisnis atau perorangan. Provider entity adalah organisasi atau orang yang menyediakan agen yang sesuai untuk menerapkan service tertentu. Requester entity adalah seseorang atau organisasi yang berkeinginan untuk menggunakan web service provider entity. Itu akan menggunakan requester agent untuk menukar pesan dengan provider agent milik provider entity.

Dalam pertukaran pesan agar berhasil, requester entity dan provider entity harus dulu sepakat menggunakan semantik dan mekanisme yang sama dalam pertukaran pesan. Semantik dalam web service adalah ekspektasi tentang perilaku (behavior) service. Ini semacam kontrak antara requester entity dan provider entity mengenai kegunaan dan konsekuensi dari interaksi tersebut. Mekanisme pertukaran pesan didokumentasikan di Web Service Description (WSD). WSD adalah spesifikasi mesin yang dapat berproses pada interface web service. menggambarkan format pesan, tipe data, protokol transpor, dan format serialisasi yang digunakan antara requester agent dan provider agent. Pada pebangunan aplikasi ini, web service berfungsi untuk menjembatani antara program frontend dan backend.

Peneliti menggunakan web service untuk ditempatkan di sebuah server sehingga aplikasi dapat menjalankan sistem sesuai yang diharapkan. Dengan menggunakan web service maka aplikasi yang dibangun dapat menyimpan data pengguna, data monitoring, dan sebagainya. Adapun metode yang digunakan untuk format pertukaran data adalah JSON. [7]

2.11. MySQL

MySQL adalah sistem manajemen database relasi yang bersifat ‘terbuka’. Terbuka maksudnya adalah MySQL boleh diunduh oleh siapa saja, baik versi kode program aslinya (source code program) maupun versi binernya (executable program) dan bisa digunakan secara (relative) gratis baik untuk dimodifikasi sesuai dengan kebutuhan seseorang maupun sebagai suatu program aplikasi komputer.

Sejarah MySQL yang merupakan hasil buah pikiran dari Michael ‘Monty’ Widenius, David Axmark, dan Allan Larson dimulai tahun 1995. Mereka bertiga kemudian mendirikan perusahaan bernama MySQL AB di Swedia. Tujuan awal ditulisnya program MySQL adalah untuk mengembangkan aplikasi web yang akan

digunakan oleh salah satu klien MySQL AB. Pada saat itu MySQL AB adalah sebuah perusahaan konsultan database dan pengembang software (masih menggunakan nama perusahaan TcX DataKonsult AB).

MySQL memiliki kinerja, kecepatan proses, dan ketangguhan yang tidak kalah dibanding database-database besar lainnya yang komersil seperti ORACLE, Sybase, Unify, dan sebagainya. Namun didalam penelitian ini peneliti menggunakan MySQL karena DBMS tersebut paling sering digunakan sehingga mudah dalam melakukan perbaikan saat terjadi kesalahan. [8]

2.12. Bahasa SQL (Struktur Query Language)

SQL (Structured Query Language) pada dasarnya adalah bahasa komputer standar yang ditetapkan untuk mengakses dan memanipulasi sistem database. Sebuah database berisi satu tabel atau lebih dan memiliki nama yang berbeda untuk setiap tabel. Setiap tabel memiliki satu kolom (field) atau lebih dan memiliki baris(record). Query digunakan untuk mengakses dan mengolah database. SQL terdiri dari empat bagian utama yaitu Data Definition Language (DDL), Data Manipulation Language (DML), Data Control Language (DCL), Data Transaction Language (DTL). Berikut adalah penjelasan dari setiap bagian tersebut :

1. *Data Definition Language* (DDL)

DDL merupakan perintah SQL yang digunakan untuk mendefinisikan, menciptakan, dan menghapus sebuah database ataupun tabel. Yang termasuk dalam perintah DDL meliputi CREATE, DROP, dan ALTER. Contoh dari perintah DDL adalah:

1. CREATE TABLE (digunakan untuk mendefinisikan sebuah tabel baru)
2. CREATE DATABASE (digunakan untuk mendefinisikan sebuah basis data yang baru)
3. CREATE INDEX (digunakan untuk membuat index)
4. CREATE VIEW (digunakan untuk membuat view)
5. DROP DATABASE (digunakan untuk menghapus basis data)
6. DROP TABLE (digunakan untuk menghapus tabel)
7. DROP INDEX (digunakan untuk menghapus index)

8. DROP VIEW (digunakan untuk menghapus view)
9. ALTER TABLE (digunakan untuk mengubah struktur suatu tabel)

2. *Data Manipulation Language* (DML)

DML merupakan perintah SQL yang digunakan untuk menambahkan, menyunting, menghapus, dan mengambil nilai dari data aktual yang didefinisikan oleh DDL. Contoh dari perintah DML adalah:

1. INSERT (digunakan untuk memasukkan nilai baru ke dalam tabel)
2. UPDATE (digunakan untuk mengubah suatu nilai pada satu atau lebih kolom)
3. DELETE (digunakan untuk menghapus suatu nilai pada tabel)
4. SELECT (digunakan untuk mengambil data dari tabel)

3. *Data Control Language* (DCL)

DCL digunakan untuk memberikan atau mencabut hak akses sumber daya tertentu didalam basis data. Pemberian hak akses dapat diterapkan pada DDL maupun DML. Contoh dari perintah TCL adalah:

1. GRANT (digunakan untuk memberikan izin)
2. REVOKE (digunakan untuk menghapus izin yang sudah ada)

4. *Data Transaction Language* (DTL)

DTL digunakan untuk mengontrol perintah dari transaksi DML hingga DDL. Contoh dari perintah DTL adalah:

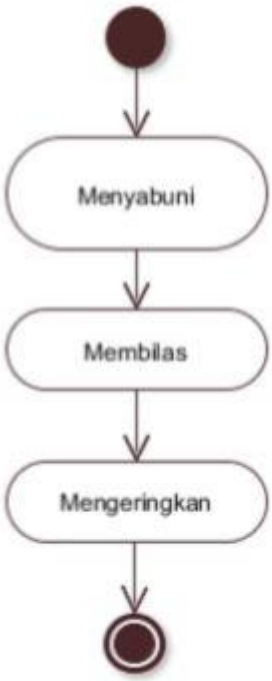
1. BEGIN (digunakan untuk memulai transaksi multistatement)
2. COMMIT (digunakan untuk mengakhiri dan menerima transaksi)
3. ROLLBACK (digunakan untuk membatalkan perubahan yang dilakukan dalam transaction atau menghapus semua data yang dimodifikasi pada awal transaction). [9]

2.13. Activity Diagram

Apabila use case menunjukkan apa yang sistem harus lakukan, berbeda halnya dengan activity diagram yang lebih membahas tentang bagaimana sistem akan mencapai tujuannya. Activity diagram menjelaskan aksi rantai pada tingkat yang lebih tinggi untuk mewakili suatu proses yang terjadi didalam sebuah sistem.

Komponen utama dari suatu activity diagram adalah action node, diwakili oleh persegi panjang bulat, yang sesuai dengan tugas yang dilakukan oleh sistem perangkat lunak. Panah dari satu action node ke action node yang lain menunjukkan aliran kontrol. Panah antara dua action node berarti bahwa setelah aksi pertama selesai, aksi kedua dimulai. Sebuah titik hitam yang solid membentuk simpul awal yang menunjukkan titik awal kegiatan. Sebuah titik hitam yang dikelilingi oleh lingkaran hitam merupakan simpul akhir menunjukkan akhir kegiatan.

Fork merupakan pemisah kegiatan menjadi dua atau lebih aktivitas bersamaan. Hal ini digambarkan sebagai bar hitam horisontal dengan satu panah menunjuk ke sana dan dua atau lebih anak panah keluar. Setiap panah keluar mewakili aliran kontrol yang dapat dieksekusi bersamaan dengan arus yang sesuai dengan panah keluar lainnya. Kegiatan bersamaan dapat dilakukan pada komputer dengan menggunakan thread yang berbeda atau bahkan menggunakan komputer yang berbeda. Sebagai contoh, activity diagram dapat dilihat pada Gambar 2.4 berikut. [10]



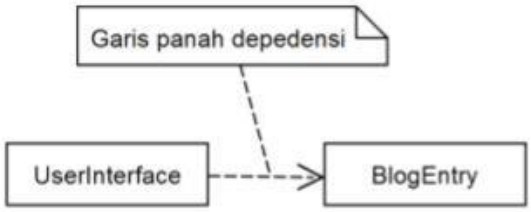
Sumber Gambar : Learning UML 2.0(2006)

Gambar 2.4 Contoh Activity Diagram Menyuci Mobil

2.13.1 *Class Diagram*

Untuk memodelkan kelas, termasuk atribut mereka, operasi, dan hubungan dan asosiasi mereka dengan kelas-kelas lain, UML menyediakan class diagram. Sebuah class diagram memberikan pandangan statis atau struktural dari sebuah sistem. Class diagram tidak menunjukkan sifat dinamis dari komunikasi antara objek dari kelas dalam diagram.

Sebuah kelas digambarkan dengan sebuah kotak yang dibagi menjadi tiga bagian. Pada bagian teratas berisi nama kelas, pada bagian tengah berisi atribut, dan pada bagian paling bawah berisi operasi-operasi yang mewakili karakteristik sebuah kelas. Sebuah atribut mengacu pada informasi yang mewakili status objek. Atribut biasanya diimplementasikan sebagai field dari sebuah kelas. Sedangkan operasi mengacu pada objek apa yang kelas dapat lakukan, biasanya diimplementasikan sebagai method dari sebuah kelas. Sebagai contoh, class diagram dapat dilihat pada Gambar 2.5 berikut.[10]



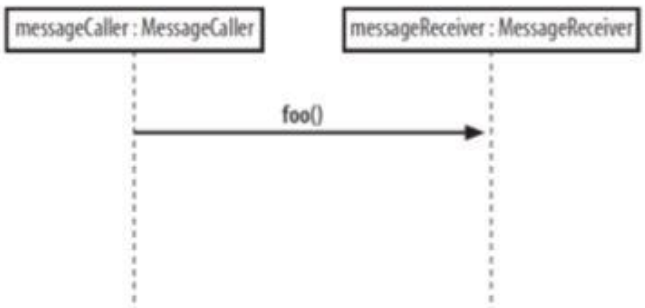
Sumber Gambar : Learning UML 2.0(2006)

Gambar 2.5 Contoh Class Diagram

Setiap atribut dapat memiliki nama, jenis, dan tingkat visibilitas. Jenis dan visibilitas adalah opsional. Jenis ini mengikuti nama dan dipisahkan dari nama dengan titik dua. Visibilitas ditunjukkan dengan sebelumnya -, #, ~, atau +, menunjukkan private, protected, package, atau public. [11]

2.13.2 *Sequence Diagram*

Sequence diagram berisi tentang urutan interaksi antara bagian-bagian didalam sebuah sistem. Menggunakan sequence diagram dapat digambarkan interaksi mana yang akan diberi trigger ketika sebagian use case dieksekusi dan dalam urutan seperti apa interaksi akan terjadi. Sequence diagram menunjukan banyak informasi lain tentang sebuah interaksi, tetapi intinya adalah menunjukan langkah sederhana dan efektif bagaimana mereka mengomunikasikan urutanurutan kejadian(events) didalam sebuah interaksi. Sebagai contoh, sequence diagram dapat dilihat pada Gambar 2.6 berikut. [11]



Sumber Gambar : Learning UML 2.0(2006)

Gambar 2.6 Contoh Sequence Diagram

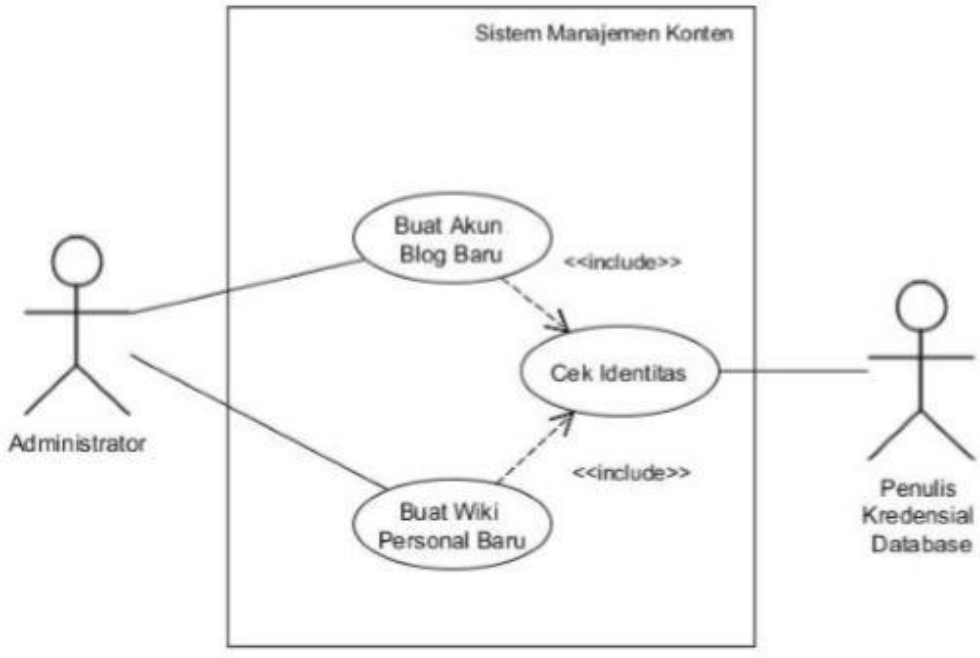
Setiap kotak pada bagian atas diagram dulunya merupakan sebuah objek. Setiap objek adalah instansi dari sebuah kelas, dan untuk menandai objek biasanya diberi garis bawah. Didalam UML 2.0, nama kotak ini disebut partisipan untuk mendeskripsikan bagian-bagian yang terlibat dialam interaksi pada sebuah sequence diagram. Di bawah setiap kotak ada garis putus-putus yang disebut lifeline objek. Sumbu vertikal dalam sequence diagram sesuai dengan waktu dimana waktu meningkat saat bergerak ke bawah. [11]

Sequence diagram menunjukkan pemanggilan metode menggunakan panah horisontal, diberi label dengan nama metode dan secara opsional termasuk parameter, jenis, dan jenis kembali. Ketika sebuah objek mengeksekusi metode maka ditampilkan bar putih, disebut bar aktivasi, di bawah lifeline objek. [11]

2.13.3 Use Case Diagram

Use case adalah situasi/case dimana sebuah sistem digunakan untuk memenuhi satu atau lebih kebutuhan pengguna. Use case dapat digunakan untuk menggambarkan fungsionalitas pada bagian-bagian yang disediakan oleh sistem. Use case menspesifikasikan apa yang harus dilakukan oleh sistem tetapi tidak menspesifikasikan apa yang seharusnya sistem tidak lakukan.

Pada Gambar 2.3 adalah contoh diagram use case. Kotak besar adalah system-boundary. Segala hal yang berada didalam kotak tersebut adalah bagian dari sebuah pengembangan sistem. Dan yang berada diluar kotak dapat dilihat aktor yang bertindak terhadap sistem. Aktor adalah entitas diluar sistem yang memberikan stimulus terhadap sistem. Aktor umunya adalah pengguna. Kemudian didalam system-boundary dapat dilihat terdapat beberapa use case. Use case berbentuk oval dengan nama kata kerja didalamnya. Para aktor dihubungkan oleh communication line kepada use case yang berhubungan. Penggunaan garis panah perlu dihindari pada communication line karena tidak seorang pun mengetahui apa maksud dari kepala panah tersebut. [12]



Sumber Gambar : Learning UML 2.0(2006) [12]

Gambar 2.7 Contoh Diagram Use case

Namun, dalam use case diagram tersebut belum menjelaskan mendetail mengenai use case yang tertera. Untuk menjelaskan use case tersebut memerlukan skenario use case. Skenario use case mendeskripsikan secara detail tentang informasi dari suatu use case yang terdapat pada use case diagram [12] . Berikut adalah tipe–tipe informasi yang digunakan dalam use case scenario dijelaskan pada Tabel 2.2 :

Tabel 2.1 Tipe Informasi pada Use Case Scenario

Use case name	Memberikan penjelasan singkat tentang nama dari use case.
Related Requirements	Daftar use case yang berhubungan dengan use case tersebut.
Goal In Context	Menjelaskan apa yang aktor coba untuk dapatkan dari use case.
Preconditions	Kondisi sistem sebelum use case dijalankan.
Successful End Condition	Kondisi sistem jika use case berhasil dijalankan.
Failed End Condition	Kondisi sistem jika use case gagal dijalankan.

Primary Actors	Aktor utama yang berpartisipasi pada use case.
Secondary Actors	Aktor yang berpartisipasi pada use case tetapi tidak menjadi yang utama.
Trigger	Event yang dipicu oleh aktor yang menyebabkan use case dijalankan.
Main Flow	Mendeskripsikan langkah yang penting pada eksekusi normal sebuah use case.
Extensions	Mendeskripsikan langkah alternatif suatu langkah di Main Flow.

Sumber Tabel: Learning UML 2.0 (2006)[12]

2.14 Analisis dan Perancangan Berorientasi Objek

Analisis dan perancangan berorientasi objek menganalogikan sistem aplikasi seperti kehidupan nyata yang didominasi oleh objek. Didalam membangun sistem berorientasi objek akan menjadi lebih baik apabila langkah awalnya didahului dengan proses analisis dan perancangan yang berorientasi objek. Tujuannya adalah mempermudah programmer didalam mendesain program dalam bentuk objek-objek beserta keterhubungan antar objek untuk kemudian dimodelkan dalam sistem nyata.

Unified Modelling Language (UML) adalah bahasa pemodelan standar untuk pengembangan perangkat lunak dan sistem. Bahasa UML dapat digunakan untuk memodelkan sistem yang menggunakan pendekatan berorientasi objek. Bahasa UML memberikan banyak fitur opsional dalam mempresentasikan semua aspek penting dalam sebuah sistem seperti menggambarkan domain masalah, desain perangkat lunak yang diinginkan, atau bahkan menggambarkan implementasi perangkat lunak yang telah selesai dibangun. Standar yang saat ini digunakan adalah UML 2.0. Didalam UML 2.0 terdapat tiga belas diagram berbeda yang dapat digunaan memodelkan perangkat lunak, diantaranya yaitu use case diagram, activity diagram, sequence diagram, class diagram, component diagram, dan deployment diagram. [13]

Pemodelan menggunakan UML digunakan peneliti karena lebih baik dalam menggambarkan sistem dari aplikasi remote monitoring smartphone Android menggunakan layanan Short Message Service (SMS) dimana setiap stimulus akan dapat menggambarkan sistem yang akan dibangun. [13]