

BAB II

LANDASAN TEORI

2.1. Teknisi

Seseorang yang memahami dan menguasai bidang dalam teknologi tertentu disebut teknisi. Teknisi banyak dijumpai dalam kehidupan sehari-hari misalnya teknisi elektro, teknisi komputer, teknisi otomotif, teknisi alat berat, teknisi ac, dan lain-lain. Untuk menjadi teknisi yang baik dibutuhkan ketelitian, keahlian dan pengalaman. Seorang teknisi mendapatkan pengalaman dari tantangan yang dihadapi dalam melakukan pekerjaannya karena itulah pengalaman yang dimiliki seorang teknisi tidaklah sama antara teknisi satu dengan teknisi lainnya. [3]

2.2. Supervisor

Mengarahkan rekan kerja bawahannya berdasarkan arahan jabatan atasannya dengan kuasa dan wewenang yang diberikan perusahaan merupakan tugas dari seorang supervisor. Tugas seorang supervisor menentukan kelancaran dan selesai tidaknya sebuah proyek, karena itu seorang supervisor diharuskan untuk berinteraksi secara langsung dengan stafnya. Supervisor bertugas menjembatani pihak manajemen dengan staf pelaksana atau staf bawah. [4]

2.3. Admin

Tugas staff administrasi secara umum dapat digambarkan sebagai staff yang membuat tujuan dari sistem atau instansi tercapai dengan mengorganisasi suatu data atau kegiatan dalam perusahaan. Administrasi adalah kegiatan menyelenggarakan,

perencanaan, mengarahkan, mengorganisir, dan mengawasi pekerjaan yang berkaitan dengan kegiatan yang ada di kantor atau instansi. [5]

2.4. Android

Android adalah sistem operasi berbasis Linux untuk perangkat pintar (smart device). Android tidak hanya dapat digunakan digunakan untuk ponsel pintar (smart phone). Terdapat dua jenis distributor sistem operasi Android. Jenis distributor android yang pertama yang mendapat dukungan penuh dari Google atau Google Mail Services (GMS) dan jenis distributor android yang kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung dari Google. [6]



Gambar 2.1. Logo Android

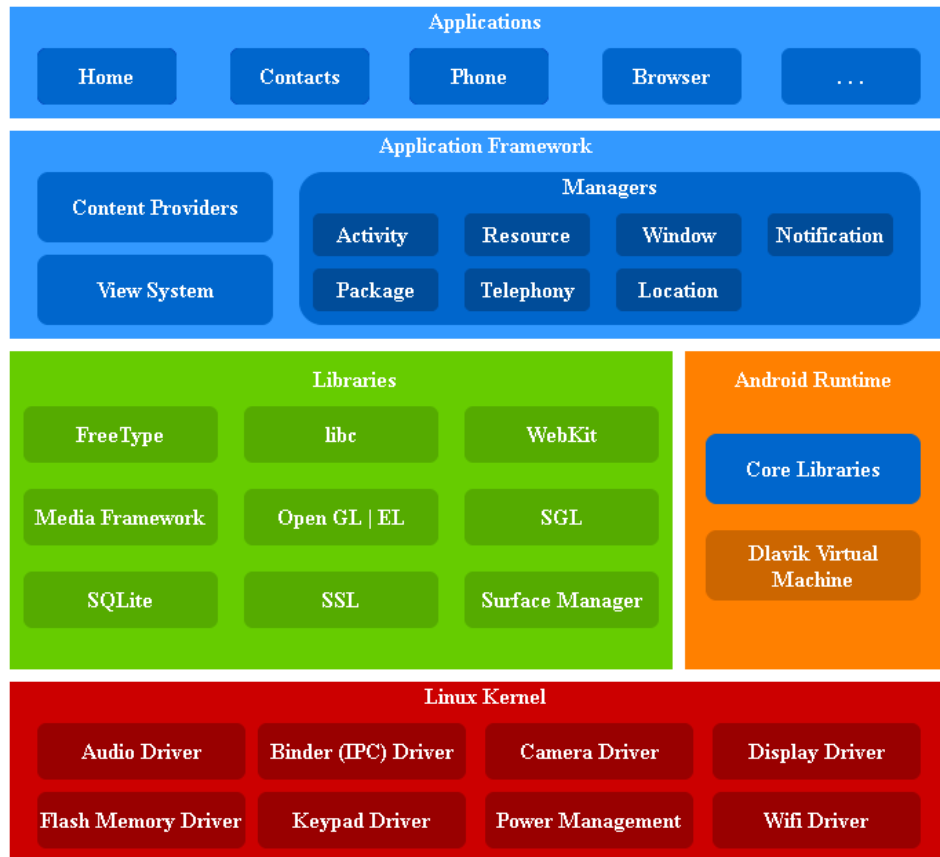
Istilah *Smart* yang diberikan pada sebuah objek elektronik menunjukkan bahwa perangkat tersebut dapat terkoneksi dengan perangkat elektronik lainnya atau terkoneksi ke internet. *Smart device* secara umum dapat diartikan sebuah perangkat elektronik yang terhubung dengan perangkat lainnya untuk mendapatkan informasi dari perangkat tersebut. *Smart device* dapat menjalankan fungsi dasar

layaknya perangkat biasa contohnya *smart tv* bisa menjalankan fungsi layaknya tv biasa. [7]

2.5. Struktur Arsitektur Android

Android merupakan platform terbuka dimana para pengembang dapat membuat aplikasi mereka sendiri. Selain platform terbuka android juga menawarkan lingkungan yang berbeda untuk pengembangannya. Android tidak membedakan antara aplikasi buatan pengembang dengan aplikasi buatan Google. API yang disediakan menawarkan akses ke hardware, data-data ponsel atau bahkan data sistem sendiri. Pengguna bahkan dapat menghapus aplikasi inti dari google dan menggantikannya dengan aplikasi buatan developer lain.

Metode object oriented digunakan saat pembangunan android, dimana elemen-elemen penyusun sistem operasinya berupa objek yang dapat kita gunakan kembali/reusable. Kita harus memahami arsitektur OS Android beserta elemennya agar dapat membuat aplikasi dengan baik. [8]



Gambar 2.2. Struktur Arsitektur Android

Secara garis besar Arsitektur Android dapat dijelaskan sebagai berikut:

2.5.1. *Application* dan *Widget*

Pada layer *Application* dan *Widget* kita berhubungan dengan aplikasi saja, biasanya kita mendownload, menginstall dan menjalankan aplikasi pada layer ini. Pada layer ini terdapat aplikasi inti seperti aplikasi SMS, kontak, browser, email, kalender, peta, dan lain-lain. Bahasa pemrograman Java seringkali digunakan untuk menulis aplikasi android. [8]

2.5.2. *Application Framework*

Pada layer *Applications frameworks* ini lah aplikasi yang akan dijalankan di sistem operasi Android dikembangkan oleh para pembuat aplikasi, karena pada layer inilah aplikasi dapat dirancang dan dibuat. Komponen yang termasuk pada *Applications Framework* antara lain adalah sebagai berikut: [8]

- *Views*
- *Notification Manager*
- *Resource Manager*
- *Activity Manager*
- *Content Provider*

2.5.3. *Libraries*

Fitur-fitur Android ada pada *layer libraries*, para pembuat aplikasi biasanya mengakses *libraries* untuk menjalankan aplikasinya. *Layer* ini berjalan di atas kernel dan meliputi berbagai library C/C++ inti seperti Libc dan SSL. Berikut beberapa library yang ada di android: [8]

- *Libraries* media untuk pemutaran media gambar, audio maupun video.
- *Libraries* Graphics SGL dan OpenGL untuk grafis 2D dan 3D
- *Libraries* SQLite untuk dukungan database
- *Libraries* SSL dan WebKit terintegrasi dengan web browser dan security.
- *Libraries* Live Webcore mencakup modern web browser dengan engine embedded web view.
- *Libraries* 3D untuk implementasi OpenGL ES 1.0 API's.

2.5.4. Android *Runtime*

Aplikasi android dapat dijalankan karena layer android *runtime* dimana dalam prosesnya menggunakan implementasi Linux. Dalvik Virtual Machine (DVM) adalah mesin bentuk dasar kerangka aplikasi android. Terdapat dua bagian dalam android Runtime, yaitu: [8]

- *Core Libraries*

Aplikasi android dibuat dengan menggunakan bahasa pemrograman Java, namun Dalvik Virtual Machine bukan merupakan Virtual Machine Java sehingga diperlukan sebuah *libraries* yang berfungsi untuk menerjemahkan bahasa Java.

- Dalvik Virtual Machine (DVM)

Virtual mesin berbasis register dioptimalkan agar berfungsi secara efisien dalam pengembangan dari Linux kernel untuk melakukan threading.

2.5.5. Linux Kernel

Layer di mana inti dari sistem operasi dari Android itu berada adalah linux kernel. Layer ini berisi file-file system yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem-sistem operasi android lainnya. [8]

2.6. SDLC (Systems Development Life Cycle)

SDLC atau System Development Life Cycle adalah proses mengembangkan suatu sistem perangkat lunak berdasarkan model dan metodologi yang sudah teruji baik dalam mengembangkan sistem-sistem perangkat lunak lainnya. Beberapa contoh model SDLC yang cukup populer dan banyak digunakan adalah *waterfall*, *fountain*, *spiral*, *rapid application development*, *prototyping*, *incremental*, *build & fix*, dan *synchronize & stabilize*. Tahap-tahap yang harus dilakukan dalam pengembangan sistem informasi adalah sebagai berikut: [9]

- Kebijakan dan perencanaan sistem (*System planning*).
- Analisa sistem (*System analysis*)
- Desain sistem (*System design*)
- Implementasi sistem (*System implementation*)
- Perawatan (*Maintenance*)

2.6.1. RAD (*Rapid Application Developments*)

Menurut [10] *Rapid Application Development* (RAD) adalah “sebuah proses perkembangan perangkat lunak sekuensial linier yang menekankan siklus perkembangan dalam waktu yang singkat”. Dalam mengembangkan sistem, RAD menggunakan metode iteratif (berulang) dimana pada tahap awal pengembangan working model (model bekerja) sistem dikonstruksi dengan tujuan menetapkan kebutuhan (requirement) pengguna dan selanjutnya disingkirkan. Sistem dapat dibangun dalam waktu 30-90 hari dengan menggunakan metode RAD, sedangkan

diperlukan waktu minimal 180 hari untuk mengembangkan sistem informasi normal.



Gambar 2.3. Rapid Application Development

Menurut [11] Terdapat 3 tahap yang terstruktur dan saling bergantung disetiap tahap dalam metode pengembangan RAD, yaitu:

1. Perencanaan Persyaratan (*Requirements Planning*):

Pada tahap ini pengguna dan analyst bertemu untuk mengidentifikasi tujuan dari aplikasi atau sistem serta mengidentifikasi syarat-syarat informasi yang ditimbulkan dari tujuan-tujuan tersebut.

2. *Design Workshop*:

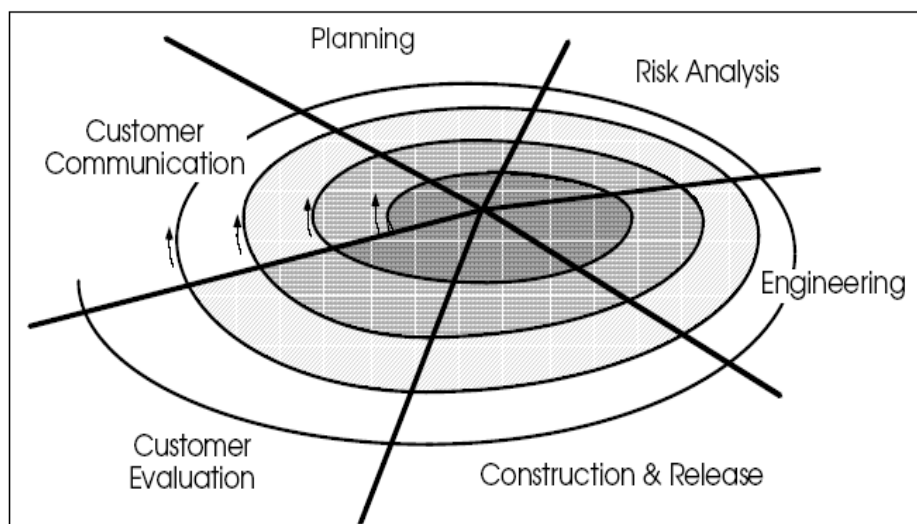
pada tahap ini *analyst* dan *programmer* bekerja membangun dan menunjukkan representasi visual desain dan pola kerja kepada . Selama tahap ini berlangsung merespon prototype yang ada dan analyst memperbaiki modul-modul yang dirancang berdasarkan respon dari . Semua perbaikan yang dilakukan berdasarkan masukan dari dilakukan pada tahap ini.

3. Penerapan (*Implementation*):

pada tahap ini desain program yang sudah disetujui dikembangkan oleh programmer. Setelah program ini selesai dikerjakan akan dilakukan pengujian untuk memastikan bahwa program yang dibuat sesuai dengan desain yang sudah disetujui. Sistem yang sudah diuji dan dipastikan tidak ada kesalahan akan mulai diperkenalkan kepada organisasi.

2.6.2. *Spiral*

Menurut [12] Model *Spiral* merupakan model pengembangan yang dihasilkan dari penggabungan Prototyping dengan aspek sistematis terkendali model air Waterfall. Manajemen resiko dalam pengembangan perangkat lunak juga diliputi oleh model spiral. Mengidentifikasi resiko utama, baik dalam hal teknis maupun manajerial, dan menentukan cara untuk mengurangi resiko untuk menjaga proses pengembangan perangkat lunak agar tetap dibawah kontrol.



Gambar 2.4. *Spiral*

2.6.3. *Waterfall*

Winston Royce memperkenalkan model pengembangan *software* ini pada tahun 70-an, model ini merupakan model klasik yang sederhana keluaran dari tahap sebelumnya menjadi masukan untuk tahap berikutnya. Model ini merupakan hasil adaptasi dari pengembangan perangkat keras, karena metodologi pengembangan perangkat lunak yang lain belum tersedia pada saat itu. Kesalahan pada proses sebelumnya dapat menyebabkan kerugian yang sangat besar dan mahal karena biaya pengembangan ulang yang membengkak yang diakibatkan proses pengembangan yang sangat terstruktur. [13]

2.7. **MySQL**

Menurut studi Statista MySQL adalah database server yang paling populer kedua setelah Oracle. MySQL merupakan *open-source software* yang gratis digunakan dibawah lisensi GNU *General Public License* (GPL) menyebabkan MySQL populer di kalangan developer.

Penggunaan SQL sebagai bahasa dasar untuk mengakses databasenya menjadikan MySQL populer dan mudah digunakan, kinerja query cepat, dan mencukupi untuk kebutuhan database perusahaanperusahaan yang berskala kecil sampai menengah, MySQL juga bersifat tidak berbayar (open source). MySQL mampu bekerja pada berbagai macam sistem operasi baik itu Windows, Linux, maupun MacOS. Selain itu bahasa pemograman seperti Java, Perl, PHP, Python, ASP dan lain sebagainya yang telah menyediakan API dan library untuk dapat mengakses MySQL. [14]

2.8. PHP

Salah satu bahasa pemrograman *open source* yang sangat cocok untuk digunakan dalam pengembangan web adalah PHP, PHP dapat ditanamkan pada sebuah skripsi HTML. PHP bekerja di sisi server, sintaks dan perintah yang diberikan oleh sisi *client* akan dikerjakan oleh server yang kemudian hasil dari proses tersebut akan ditampilkan disisi *client*. Karena proses PHP dilakukan di sisi server kode program yang ditulis dalam PHP tidak akan terlihat oleh *client* sehingga keamanan halaman web lebih terjamin. [15]

Beberapa database yang sudah didukung akses PHP antara lain MySQL, PostgreSQL, infomix, dan MicrosoftSQL Server. PHP didukung oleh banyak web server, antara lain IIS, Apache, Fhttp, phttp, AOsriver, Lighttpd, PWS hingga Xitami dengan konfigurasi yang relative mudah. [16]

2.9. JSON

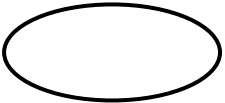


JavaScript Object Notation atau biasa disebut JSON adalah suatu format ringkas pertukaran data komputer yang mudah dipahami manusia dan diterjemahkan dan dibuat(generate) oleh mesin. Sesuai namanya JSON mempunyai format berbasis teks dengan berdasarkan pada subset bahasa pemograman JavaScript. Walaupun begitu penggunaan JSON sudah didukung oleh banyak bahasa pemograman seperti C, C++, C#, Java, JavaScript, PHP, dan Perl hal ini menjadikan JSON ideal sebagai bahasa pertukaran data. [17]




JSON diciptakan dan dipopulerkan oleh DouglasCrockford pada tahun 2001 dan mulai digunakan untuk *web service* oleh Yahoo pada Desember 2005 dan Google pada Desember 2006. Sebagai format pertukaran data komputer JSON merupakan alternatif pengganti XML karena strukturnya yang lebih ringkas dan sederhana.

2.10. Use Case Diagram

Hal-hal yang dapat dilakukan oleh aktor dalam menyelesaikan sebuah pekerjaan direpresentasikan oleh *use case* diagram. Behavior (kelakuan) sistem informasi yang akan dibuat dimodelkan oleh diagram *use case*. Interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat dideskripsikan oleh sebuah *use case*. Secara singkat, *use case* digunakan untuk mengetahui apa saja fungsi yang ada di dalam sebuah sistem dan siapa saja yang memiliki hak untuk menggunakan fungsi-fungsi tersebut. [18]

Tabel 2.1. Komponen pada Use Case Diagram

Simbol	Keterangan
	<i>Use Case</i> : fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan actor.
	<i>System Boundary</i> : Kotak yang menetapkan cakupan sistem untuk use case. Semua use case di luar kotak akan dianggap di luar cakupan sistem itu.
	Aktor: orang, proses, atau sistem lain yang berinteraksi dengan sistem yang dibuat menggunakan <i>use case</i> .

Simbol	Keterangan
	Asosiasi: Asosiasi antara aktor dan <i>use case</i> , bukan mengindikasikan data melainkan mengindikasikan siapa atau apa yang meminta interaksi dengan <i>use case</i> .
<<extend>> 	<i>Extend</i> : digunakan untuk menunjukkan relasi use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan.
<<include>> 	<i>Include</i> : digunakan untuk menunjukkan relasi use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya

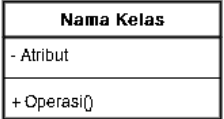





Sumber: [19]

2.11. Class Diagram

Penjelasan detail dari tiap kelas dan hubungan antar kelas di dalam desain model dari suatu sistem digambarkan dalam sebuah class diagram, selain itu class diagram juga menunjukkan tanggung jawab dan aturan entitas yang menentukan perilaku sistem. Atribut dan operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan juga ditunjukkan dalam class diagram [18]. Berikut ini merupakan Elemen *Class Diagram*:

- *Class* (Nama)
- *Attributes*
- *Operations*
- *Relationships*

Tabel 2.2. Komponen pada Class Diagram




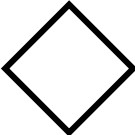

Simbol	Keterangan
	Kelas pada struktur sistem.
	Asosiasi: Relasi antar kelas dengan makna umum.
	Asosiasi Berarah: relasi antar kelas dengan makna kelas yang satu digunakan dengan kelas yang lain.
	Generalisasi: relasi antar kelas dengan makna generalisasi-spesialisasi.
	Kebergantungan: relasi antar kelas dengan makna kebergantungan antar kelas.
	Agregasi: relasi antar kelas dengan makna semua bagian (<i>whole part</i>).

Sumber: [19]

2.12. Activity Diagram

Work flow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis digambarkan dalam *activity* diagram. Perlu diperhatikan bahwa *activity* diagram bukan menggambarkan apa yang dilakukan aktor melainkan menggambarkan aktivitas sistem, jadi aktivitas dapat dilakukan oleh sistem. [18]

Tabel 2.3. Activity Diagram

Simbol	Keterangan
	<i>Initial State</i> : Menggambarkan awal dimulainya suatu aliran aktivitas
	<i>Final State</i> : Menggambarkan berakhirnya suatu aliran aktivitas
	<i>Action</i> : Menggambarkan aktivitas atau proses yang dilakukan dalam suatu aliran aktivitas
	<i>Decision</i> : Menggambarkan pilihan kondisi atau cabang-cabang aktivitas tertentu
	<i>Transition</i> : Berguna untuk Menghubungkan satu komponen dengan komponen lainnya

Sumber: [19]

2.13. Penelitian Terdahulu

Tabel 2.4. Penelitian Terdahulu

No	Judul	Permasalahan	Hasil	Metode
1	<p>Perancangan Klaim Untuk Human Resources Development System Pada PT Protonema Berbasis Android</p> <p>Nama Penulis: Ayu Delima, Rudi Hermawan, & Harmi Ibnu Djafar</p> <p>Tahun Penulisan: 2020</p> <p>Jurnal: JRAMI (Jurnal Riset dan Aplikasi Mahasiswa Informatika) Vol 01 No 01</p>	<p>Dikarenakan banyaknya jumlah karyawan yang bekerja di PT Protonema, pihak perusahaan merasa kewalahan dan memerlukan waktu yang lama untuk memproses klaim yang diajukan karyawannya seperti klaim kesehatan, klaim lembur, dan detail data lainnya. Semua proses tersebut masih dilakukan secara manual sehingga setiap proses yang dilakukan masih memerlukan waktu yang lama.</p>	<p>Merancang dan membangun aplikasi yang dapat mempermudah dan mempercepat klaim yang diajukan karyawan.</p>	<p>Prototyping</p>

No	Judul	Permasalahan	Hasil	Metode
2	<p>Perancangan Aplikasi Penerimaan Dan Penugasan Personil Pengamanan PT Siap Andalan Paramitra Berbasis Android</p> <p>Nama Penulis: Agus Kodir Arifin Tahun Penulisan: 2019 Jurnal: Jurnal String Vol. 3 No. 3</p>	<p>PT Siap Andalan Paramitra bergerak di bidang penyedia jasa pengamanan dan menghadapi permasalahan dalam penyampaian informasi antara calon pelamar dan bagian penerimaan karyawan baru serta kesulitan dalam memberikan informasi penugasan bagi karyawan yang sudah diterima.</p>	<p>Membangun aplikasi yang dapat memudahkan bagian penerimaan karyawan baru untuk memproses data pelamar, serta memudahkan karyawan yang telah diterima untuk melihat informasi penugasan.</p>	<i>Grounded Research</i>
3	<p>Perancangan Aplikasi Pelaporan Berbasis Android Pada Komunikasi Jaringan Data 4G</p> <p>Nama Penulis: R. Deiny Mardian, Suhartati Agoes, & Revi S. Riffany Tahun Penulisan: 2018 Jurnal: SNTI VI</p>	<p>Laporan nilai penjualan suatu produk/kinerja suatu cabang masih berbentuk fisik hal ini menyebabkan laporan sulit diakses bagi personil yang membutuhkan laporan tersebut karena laporan harus dikirim fisiknya.</p>	<p>Membangun aplikasi yang dapat mempermudah dan mempercepat personil untuk membuat dan mengakses laporan yang dibutuhkan</p>	RAD

Penelitian yang berjudul "Perancangan Klaim Untuk Human Resources Development System Pada PT Protonema Berbasis Android" yang di tulis oleh Ayu Delima, Rudi Hermawan, & Harmi Ibnu Djafar membahas permasalahan yang dihadapi PT Protonema, yaitu perusahaan tersebut kewalahan dalam memproses klaim yang diajukan karyawannya seperti klaim kesehatan, klaim lembur, dan lain-lain. Permasalahan ini serupa dengan permasalahan yang dihadapi PT Viros Prime Solution hanya saja pada PT Viros, yang kesulitan adalah teknisi yang ingin mengajukan reimbursement. Penelitian ini mengadopsi fitur klaim menggunakan aplikasi hanya saja disesuaikan dengan permasalahan yang dihadapi oleh PT Viros Prime Solution.

Penelitian yang berjudul "Perancangan Aplikasi Penerimaan Dan Penugasan Personil Pengamanan PT Siap Andalan Paramitra Berbasis Android" yang di tulis oleh Agus Kodir Arifin membahas permasalahan yang dihadapi oleh PT Siap Andalan Paramitra, yaitu perusahaan tersebut kesulitan dalam memberikan informasi penugasan bagi karyawan. PT Viros Prime Solution juga menghadap permasalahan yang serupa, supervisor hanya memberikan pekerjaan kepada teknisi secara lisan kepada teknisi hal ini dapat menyulitkan supervisor karena harus mencatat ulang pekerjaan apa yang diberikan ke teknisi dan hal ini juga dapat menyulitkan teknisi karena detail pekerjaan yang diberikan secara lisan dapat menyebabkan miskomunikasi. Solusi yang dibuat oleh penelitian tersebut dapat digunakan sebagai inspirasi dalam penelitian ini agar supervisor memiliki sarana untuk memberikan pekerjaan kepada teknisi secara terstruktur.

Penelitian yang berjudul "Perancangan Aplikasi Pelaporan Berbasis Android Pada Komunikasi Jaringan Data 4G" yang di tulis oleh R. Deiny Mardian, Suhartati Agoes, & Revi S. Riffany membahas masalah pengiriman laporan dan kesulitan dalam mengakses laporan karena laporan masih berbentuk fisik (tertulis/print). Penelitian tersebut dijadikan sebagai acuan untuk membangun fitur laporan teknisi dalam penelitian ini, karena teknisi pada PT Viros Prime Solution masih mengirimkan laporan dengan menggunakan email dan whatsapp sehingga supervisor kesulitan dalam menyimpan dan mengakses laporan karena laporan yang dikirimkan melalui email dan whatsapp harus dipindahkan kedalam excel.