BAB II
LANDASAN TEORI

A. Informasi

1. Pengertian Sistem Informasi

Informasi adalah pengelolahan suatu data mentah menjadi suatu informasi yang dipahami oleh pengguna. Sistem informasi terdiri dari kata sistem dan informasi. Menurut Jogiyanto (1991:1) sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu". Sistem merupakan bagian – bagian yang saling terkait dan membentuk kesatuan untuk mencapai tujuan. Bagian – bagian dalam sistem bekerja pada cakupan khusus agar mecapai tujuan dari sistem secara menyeluruh. Sistem dapat terdiri dari beberapa cakupan seperti orang, informasi, dan lain – lain.

Informasi merupakan bagian penting dalam pengambilan keputusan dan digunakan dalam suatu manajemen modern. Informasi menurut Jogiyanto (1999:692) merupakan hasil dari pengolahan data dalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanya yang menggambarkan suatu kejadian – kejadian (*event*) yang nyata (*fact*) yang digunakan untuk pengambilan keputusan". Informasi adalah pengolahan suatu data mentah menjadi suatu informasi yang dipahami oleh pengguna. Sistem informasi menurut A. Leitch dan Davis (1983:6) merupakan suatu sistem yang terdiri dari kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan laporan yang diperlukan. Sistem informasi merupakan komponen yang saling bekerja sama dan menghasilkan informasi yang akurat, dan tepat guna. Sistem informasi disusun untuk mempermudah berbagai pihak dalam memperoleh informasi.

2. Komponen Sistem Informasi

Menurut Jogiyanto (2005:4) komponen sistem informasi yaitu bagian-bagian dari sistem yang saling bekerja sama dan berinteraksi membentuk

kesatuan. Dalam buku Al-Bahra Bin Ladjamudin (2005:14) komponen sistem terdiri dari lima macam, anatara lain :

1) Perangkat keras

   Perangkat keras yaitu perangkat komputer yang fisiknya dapat dipegang. Perangkat keras meliputi seperangkat komputer yang terdiri dari *Unit Personal Sentral* (CPU), monitor, perangkat *input*, perangkat *output*, serta perangkat proses yaitu perangkat penyimpanan data.

2) Perangkat Lunak

   Perangkat lunak yaitu perangkat yang berada dalam sistem komputer dan biasanya bersifat sebagai aplikasi.

3) File

   File merupakan arsip dalam sistem yang terdiri dari program dan data. definisi *file* lainnya ialah arsip ataupun data yang tersimpan di dalam komputer.

4) Prosedur

   Prosedur yaitu komponen fisik yang digunakan sebagai panduan dalam pengoprasian sistem atau serangkaian aksi yang spesifik, tindakan atau operasi yang harus dijalankan atau dieksekusi dengan cara yang baku (sama) agar selalu memperoleh hasil yang sama dari keadaan yang sama.

5) Pemakai

   Pemakai yaitu pengguna sistem. Pengguna sistem meliputi operator, personalia, sistem analis, admin, dan lainnya.

Komponen sistem informasi menurut John Burch dan Gray Grudnistski dalam buku Jogiyanto (2005:12) terdiri dari beberapa blok :

1) Blok masuk input

   Blok masuk input yaitu sebuah bagian yang mangarut data masukan seperti motode dan media yang akan digunakan.

2) Blok model

   Blok model yaitu sebuah bagian yang terdiri dari kombinasi

prosedur, logika dan model matematik untuk memanipulasi data masukan sehingga sesuai dengan data keluaran yang diinginkan.

3) Blok keluaran produk

Blok keluaran produk yaitu bagian yang mengatur tentang sistem keluaran baik berupa informasi maupun data yang berkualitas dan berguna bagi pemakai.

4) Blok teknologi

Blok teknologi yaitu bagian yang mengendalikan sistem, dimulai dari menerima masukan, menjalankan model, menyimpan data, hingga menampilkan keluaran.

5) Blok basis data

Blok basis data yaitu bagian dari sistem yang mengatur tentang semua data baik yang saling berhubungan ataupun tidak, lalu menyimpannya kedalam komputer dan memanipulasi didalam sistem.

6) Blok kendali

Blok kendali yaitu bagian dan sistem yang mengatur tentang upaya-upaya yang dilakukan guna mengendalikan sistem sehingga dapat mencegah kerusakan atau kesalahan sistem lainnya.

3. Karakteristik Sistem Informasi

Menurut Jogiyanto (2005:4) sistem informasi memiliki karakteristik atau sifat tertentu, seperti komponen, batasan sistem, lingkungan, penghubung, masukan, keluaran, proses, dan tujuan.

1) Komponen (*Components*)

Sistem informasi yang bak harus berdasarkan karakteristik sistem, sistem informasi terdiri dari beberapa komponen yang saling berinteraksi dan bekerjasama.

2) Batasan Sistem (*Boundary*)

Setiap komponen sistem memiliki batasan fungsi antar komponen setiap sistem informasi memiliki batasan sistem dengan sistem lainnya. Batasan sistem menjadikan sistem informasi bersifat unik.

3) Lingkungan (*Environments*)

Lingkungan sistem terdiri dari lingkungan dalam dan luar sistem berkaitan dengan batasan sistem. Batasan sistem sebagai pemisah antara sistem dan lingkungan.

4) Penghubung (*Interface*)

Penghubung atau *interface* merupakan gambaran nyata dari sistem. Penghubung digunakan sebagai jembatan dari tiap komponen atau sub sistem.

5) Masukan (*Input*)

Masukan sistem terkait dengan data yang digunakan dalam mengembangkan sistem. Setiap masukan akan diproses untuk menghasilkan keluaran.

6) Keluaran (*Output*)

Keluaran merupakan hasil dari masukan yang telah diproses. Setiap keluaran akan menghasilkan informasi yang berguna dan sesuai dengan kebutuhan masukan. Keluaran dirancang berdasarkan keburugan pemakai sistem.

7) Pengolah (*Process*)

Proses sistem merupakan fungsi yang terdapat pada sistem untuk menjalankan sistem dimulai dari masukan sehingga menghasilkan keluaran yang diinginkan.

8) Tujuan (*Goals*)

Tujuan atau sasaran merupakan penentu keberhasilan dari sistem. Setiap sistem memiliki tujuan pencapaian. Sistem dikatakan berhasil jika sudah sesuai dengan tujuan sistem, atau jika masukan dan keluaran yang dihasilkan sama.

B. Presensi Mahasiswa

Salah satu kegiatan dosen saat berada di kelas adalah mendata kehadiran mahasiswa atau biasa disebut presensi. Menurut kamus Besar Bahasa Indonesia (KBBI) absensi berarti ketidakhadiran yang berasal dari kata absent (tidak hadir). Kehadiran mahasiswa merupakan salah satu penilaian utama dalam proses pembelajaran. Menurut Permenristekdikti Nomor 44 Tahun 2015 tentang Standar

Nasional Pendidikan dalam Standar Proses Pembelajaran diatur tentang semester yang berisi bahwa "semester merupakan satuan waktu kegiatan pembelajaran efektif selama paling sedikit 16 (enam belas) minggu termasuk ujian tengah semester dan ujian akhir semester". Selain itu, berdasarkan Standar Penjaminan Mutu Internal (SPMI) dalam Standar Pendidikan yang berlaku di IKIP PGRI Pontianak menyebutkan bahwa kehadiran mahasiswa menjadi syarat untuk dapat mengikuti ujian akhir semester. Jumlah minimal kehadiran mahasiswa yaitu 75% kehadiran mahasiswa. Jumlah kehadiran dosen juga diatur berdasarkan jumlah sks yang diajarkan.

Berdasarkan peraturan yang berlaku baik secara nasional maupun di IKIP PGRI Pontianak, maka kehadiran mahasiswa merupakan salah satu bagian penting dan harus diperhatikan oleh dosen, pimpinan program studi, maupun pimpinan perguruan tinggi. Kehadiran mahasiswa merupakan salah satu komponen dalam penilaian akhir mata kuliah selain tugas, ujian tengah semester, dan ujian akhir semester dan tertuang dalam setiap Rencana Pembelajaran Semester (RPS) mata kuliah. Kehadiran mahasiswa dalam perkuliahan harus terdokumentasi dengan baik yaitu dengan dilakukan pencatatan dan pelaporan presensi dari dosen maupun mahasiswa. Selain itu, kehadiran mahasiswa digunakan sebagai bahan evaluasi kepuasan mahasiswa terhadap mata kuliah dan tolak ukur bagi dosen dalam memberikan materi yang lebih baik kedepannya.

Beberapa penelitian yang telah ditetapkan menunjukkan bahwa kehadiran mahasiswa selama perkuliahan memiliki korelasi positif dengan kinerja akademik mereka (Dobkin, 2010; Westerman, 2011; Chau dan Kuo, 2012) dan menurut Amelia dan Susi (2014:39) mengatakan bahwa 52,70% nilai akhir mahasiswa ditentukan berdasarkan kehadiran mahasiswa lebih dari 70%. Berdasarkan hal tersebut dapat diketahui bahwa kehadiran mahasiswa dapat menjadi faktor utama untuk mendapatkan nilai akhir yang memuaskan.

Setiap instansi perusahaan, pemerintah maupun pendidikan pasti membutuhkan suatu sistem presensi untuk memulai pekerjaan atau pembelajaran. Presensi merupakan bagian peranan penting dalam instansi pendidikan. Dimana presensi merupakan salah satu penunjang utama yang dapat mendukung dan

memotivasi setiap kegiatan yang dilakukan di dalamnya. Seperti halnya sistem presensi di IKIP PGRI Pontianak masih menggunakan metode konvensional, cara ini sangatlah rawan bagi suatu Lembaga Pendidikan karena tingkat kedisplinan yang tidak dapat di kontrol dan dapat di salah gunakan oleh orang yang tidak bertanggung jawab dengan memanipulasi tanda tangan atau titip presensi jika dosen yang bersangkutan tidak melakukan pemeriksaan pada lembar kehadiran yang telah diisi oleh mahasiswa. Selain itu, dapat juga terjadi kehilangan atau rusaknya lembar kehadiran yang akan mengakibatkan hilangnya informasi kehadiran perkulihan yang telah dilaksanakan dan kerugian lainnya mungkin muncul pada sistem presensi manual adalah rekapitulasi data yang masih memakan banyak waktu dan tenaga.

Data presensi disimpan di *Firebase*. *Firebase* memiliki fitur sangatlah banyak dan membuat penyimpanan data lebih baik di bandingkan basis data lainnya. Dikarenakan *database* yang disinkronkan secara *realtime* ke setiap klien yang terhubung sehingga klien dapat menerima *update* data terbaru secara otomatis. Tidak hanya itu *Firebase* memiliki fasilitas untuk mengembangkan aplikasi berkualitas tinggi seperti halnya memiliki *Authentication* untuk layanan masuk ke suatu sistem seperti fitur daftar, masuk bahkan lupa kata sandi dan lainnya. *Realtime Database* untuk klien menerima *update* data terbaru secara otomatis sedangkan *Cloud Firestore Database* memiliki fitur kueri yang lebih lengkap dan skala yang lebih baik dibandingkan *realtime database* lainnya. *Cloud Storage Firebase* untuk penyimpanan foto maupun video atau *file* lainnya. ML Kit *for Firebase* untuk mengenali teks, wajah, kode batang, melabeli gambar, mendeteksi dan melacak objek bahkan mengenali bangunan terkenal dan masih banyak lagi fasilitas yang di sediakan oleh *Firebase*.

C. Internet

Internet adalah suatu jaringan komputer yang pertama kali dibentuk oleh Departemen Pertahanan Amerika Serikat di tahun 1969, dengan proyek ARPA yang disebut ARPANET. ARPA NET kekepanjangan dari *Advanced Research Project Agency Network* yang mendemonstrasikan cara suatu *hardware* dan

software komputer memiliki basis UNIX dengan berkomunikasi dalam jarak yang jauh dengan saluran telepon.

Internet adalah sebuah jaringan komputer yang saling terhubung dengan menggunakan suatu sistem standar global *transmission control* protocol/internet protocol suite (TCP/IP) yang digunakan sebagai protokol pertukaran paket dalam melayani miliaran pengguna yang terdapat di seluruh dunia. Internet merupakan kependekan dari *interconnected network*. Internet juga dapat diartikan sebagai jaringan komunikasi global yang terbuka dan menghubungkan jutaaan atau milyaran jaringan komputer dengan berbagai tipe dan jenis, dengan menggunakan tipe komunikasi misalnya telepon, satelit, dan sebagainya. Fungsi Internet secara sederhana adalah sebagai berikut:

1. Sebagai media komunikasi
2. Sebagai salah satu tempat untuk akses informasi
3. Berbagi sumber daya atau data
4. Dapat menyiarkan dan mengakses secara langsung baik itu berita dan bertukar data dengan internet *online* ke seluruh dunia.

D. Android

Android adalah sistem operasi mobile bersifat *open source* yang dikembangkan *Google Corporation* yang merupakan perusahaan mesin pencari terkemuka di dunia. Para pengembang dapat membuat aplikasi dengan menggunakan *platform* android untuk berbagai perangkat bergerak. Android menjadi sistem operasi yang sangat populer karena tingkat efektivitas dan efisiensinya yang lebih baik dibandingkan dengan program sejenis lainnya, sehingga Android juga populer digunakan untuk kepentingan pendidikan karena kemudahan dan fleksibilitasnya. Pembelajaran yang mengadopsi sistem dan perangkat *mobile* selanjutnya dikenal dengan istilah *mobile learning*. Android sendiri dapat digunakan sebagai sarana belajar mandiri bagi siapa pun, baik di sekolah maupun di rumah. Hal ini menegaskan bahwa *mobile learning* berbasis Android menawarkan kesempatan yang bagi siapapun untuk dapat mengakses pembelajaran secara mudah dan menyenangkan. Selain itu, Dixit (2014:2) mengatakan bahwa Android merupakan perangkat seluler yang digunakan untuk

perangkat *mobile* yang meliputi sistem operasi, *middleware*, dan aplikasi inti. Android berbasis sistem operasi *Linux* didesain untuk perangkat mobile layar sentuh seperti *smartphone* dan komputer *tablet*.

Pada saat ini Android telah diminati banyak masyarakat sebagai media komunikasi. Kelebihan Android dibandingkan ponsel lain seperti yang diungkapkan oleh Kusuma (2011:10-12) yaitu:

1. *Multitasking*

   Multitasking memiliki arti bahwa sistem Android mampu menjalankan beberapa aplikasi sekaligus yang tidak terbatas, baik aplikasi-aplikasi yang berasal dari bawaan sistem atau tambahan dari Android *Marketplace*. Seperti contohnya adalah seseorang dapat mendengarkan musik sambil *browsing* internet, dan juga menerima notifikasi dapat dilakukan.

2. *Home screen fleksibel*.

   Home screen merupakan jendela utama sistem, di mana segala notifikasi dapat dipantau. *Homescreen* dapat digunakan untuk menaruh *shortcut* aplikasi-aplikasi yang sering digunakan pengguna. Selain itu Android menyediakan tempat bagi berbagai *widget*.

3. Banyak pilihan piranti

   Maksudnya adalah vendor pendukung sistem ini banyak. Jadi pilihan perangkat yang bisa digunakan sangat beragam dan juga dengan harga yang bervariasi. Rata-rata Android menggunakan layar sentuh dengan ukuran mulai 2,8 inci. Ada Android yang khusus dibuat untuk navigasi maupun multimedia, namun ada pula berwujud *tablet* atau *notebook*.

4. Modifikasi sistem.

   Selain beberapa kelebihan di atas, Android memberikan banyak kebebasan dalam hal modifikasi sistem. Beberapa hal yang bisa dilakukan adalah *rooting* dan modifikasi ROM sistem.

5. Pengesetan yang mudah.

   Android telah dikembangkan sejak lama dan siap dipakai dengan mudah. Pengesetan untuk keperluan sehari-hari menyesuaikan dengan aktivitas pengguna dapat dilakukan dengan mudah tanpa perlu banyak

mengutak-atik. Dibandingkan dengan sistem operasi *mobile* lainnya, Android memiliki beberapa kelebihan, seperti dukungan format audio yang kaya, dukungan *multitouch*, banyaknya pilihan aplikasi, terlebih yang gratis dan *open source*. Kelebihan Android lainnya adalah dukungan multimedia yang komplit dan beragam.

Rogers dan kawan-kawan (2009:3) menyebutkan bahwa Android memiliki potensi untuk menghilangkan hambatan untuk sukses dalam pengembangan dan penjualan perangkat lunak aplikasi dari perangkat *mobile* generasi baru. Seperti halnya PC dan *Machintosh* yang menciptakan pasar untuk perangkat lunak *desktop* dan *server*, maka Android dengan menyediakan aplikasi berbasis *mobile* akan menciptakan pasar untuk aplikasi *mobile* dan hal ini merupakan kesempatan bagi pengembang aplikasi untuk mendapatkan banyak keuntungan. Perkembangan teknologi informasi yang semakin pesat khususnya teknologi berbasis *mobile* telah memberi banyak kemudahan untuk melakukan berbagai aktifitas. Pada era dahulu orang melakukan komunikasi jarak jauh dengan menggunakan telepon. Seiring perkembangan zaman, teknologi telepon telah banyak berkembang. Seperti contohnya Android, yang saat ini telah dikenal dan digunakan oleh hampir seluruh masyarakat di penjuru dunia.

E. Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu (*Integrated Development Environment/IDE*) resmi untuk pengembangan aplikasi Android, yang didasarkan pada IntelliJ IDEA. Selain sebagai editor kode dan fitur developer IntelliJ yang andal, Android Studio menawarkan banyak fitur yang meningkatkan produktivitas Anda dalam membuat aplikasi Android, seperti:

1. Sistem build berbasis *Gradle* yang fleksibel
2. *Emulator* yang cepat dan kaya fitur
3. Lingkungan terpadu tempat Anda bisa mengembangkan aplikasi untuk semua perangkat Android
4. Terapkan Perubahan untuk melakukan push pada perubahan kode dan *resource* ke aplikasi yang sedang berjalan tanpa memulai ulang aplikasi

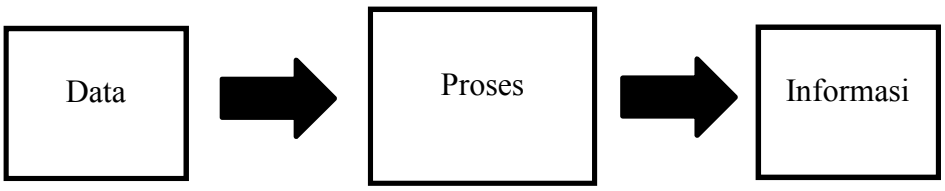
5. *Template* kode dan integrasi *GitHub* untuk membantu Anda membuat fitur aplikasi umum dan mengimpor kode sampel
6. *Framework* dan fitur pengujian yang lengkap
7. Fitur lint untuk merekam performa, kegunaan, kompatibilitas versi, dan masalah lainnya
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk *Google Cloud Platform*, yang memudahkan integrasi *Google Cloud Messaging dan App Engine*

F. Database

1.Pengertian *Database*

Menurut Abdul Kadir (2006:2) secara sederhana *database* (basis data) dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan cepat. Dalam hal ini, pengertian akses dapat mencangkup pemerolehan data maupun pemanipulasian data, seperti menambah dan menghapus data.

Menajemen modern mengikutsertakan informasi sebagai sumber daya penting yang setara dengan sumber daya manusia. Uang, mesin, dan material. Informasi adalah suatu bentuk penyajian data yang melalui mekanisme pemrosesan, yang berguna bagi pihak tertentu, misalnya manajer. Bagi pihal manajemen, informas merupakan bahan untuk pengambilan keputusan.



Gambar 2.1. Data dan Informasi

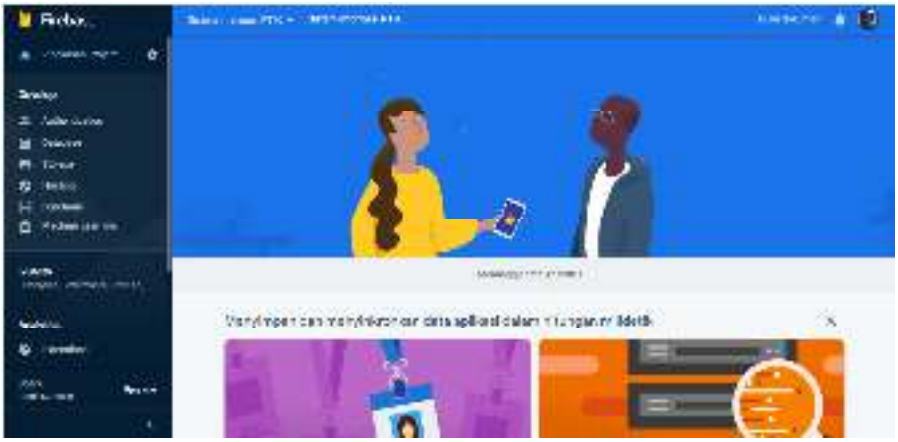
Dengan adanya komputer, data dapat disimpan dalam media pengingat yang disebut *hard disk*. Dengan menggunakan media ini kehadiran kertas yang digunakan untuk menyimpan data dapat dikurangi. Selain itu, data menjadi lebih baik cepat untuk diakses terutma kalau dikemas dalam bentuk *database*.

Manfaat *database* banyak dijumpai di sekeliling kita. ATM (Anjungan Tunai Mandiri) merupakan sebuah contoh teknologi informasi yang pada dasarnya memanfaatkan *database*, yang memungkinkan seseorang bisa mengambil uang di mana saja dan kapan saja. Didalam *database* tersimpan data yang menyangkut rekening nasabah. *Password* yang valid untuk nasabah, dan juga saldo tabungan nasabah. Aplikasi *database* yang lain dapat dijumpai di pada toko-toko swalayan, perpustakaan, dan bahkan pada internet. Sebagai gambaran, dengan menggunakan aplikasi web, seseorang bisa melihat buku-buku pada situs-situs toko buku *online* dengan hanya memasukkan kata kunci tertentu seperti *databases* atau *Access*. Aplikasi tersebut akan mencocokkan dengan *database* yang tersedia dan kemudian menampilkan judul-judul buku berserta atribut lainnya (nama pengarang, ISBN, dan sebagainya) ke layar komputer pemakai.

DBMS singkatan dari *Database Management System.* DBMS merupakan perangkat lunak atau program kemputer yang dirancang secara khusus unruk memudahkan pengelolaan database. Salah satu macam DBMS yang populer dewasa ini berupa RDBMS (*Relational Database Management System*), yang menggunakan model basis data relasional atau dalam bentuk tabel- tabel yang saling terhubungkan.

MySQL merupakan salah satu contoh produk RDBMS yang sangat populer di lingkungan Linux, tetapi juga tersedia pada *Windows*. Banyak situs *Web* yang menggunakan *MySQL* sebagai *database server* (*server* yang melayani permintaan akses terhadap *database*).

Firebase memiliki fitur sangatlah banyak dan membuat penyimpanan data lebih baik di bandingkan basis data lainnya. Tampilan pada *Firebase* pada *project* seperti di bawah ini yang memiliki fitur atau fasilitas yang bermanfaat untuk aplikasi android yang kita buat.



Gambar 2.2 Tampilan *Firebase*

Untuk membuat pengguna *login* ke aplikasi, dapatkan kredensial autentikasi dari pengguna terlebih dahulu. Kredensial ini dapat berupa alamat *email* dan sandi pengguna, atau token *Auth* dari penyedia identitas gabungan. Kemudian, teruskan kredensial ini ke *Firebase Authentication* SDK (*Software Development Kit*). Layanan *backend* kami selanjutnya akan memverifikasi kredensial tersebut dan menampilkan *respons* ke klien.

Setelah berhasil *login*, Anda dapat mengakses informasi profil dasar pengguna dan mengontrol akses pengguna ke data yang disimpan di produk *Firebase* lainnya. Anda juga dapat menggunakan token autentikasi yang disediakan untuk memverifikasi identitas pengguna di layanan *backend* yang dimiliki.

Simpan dan sinkronkan data Anda dengan *database cloud NoSQL*. Data disinkronkan pada semua klien secara *realtime* dan tetap tersedia meski aplikasi Anda *offline*. *Firebase Realtime Database* adalah *database* yang di-*host* (jaringan) di *cloud*. Data disimpan sebagai JSON (*JavaScript Object Notation*) dan disinkronkan secara *realtime* ke setiap klien yang terhubung. Ketika Anda membuat aplikasi lintas-*platform* dengan SDK Android, iOS, dan *JavaScript*, semua klien akan berbagi sebuah *instance* (struktur proses dan *memory* yang menjalankan sistem *database*) *Realtime Database* dan menerima *update* data terbaru secara otomatis.

Firebase Realtime Database memungkinkan Anda untuk membuat aplikasi kolaboratif dan kaya fitur dengan menyediakan akses yang aman ke

database, langsung dari kode sisi klien. Data disimpan di *drive* lokal. Bahkan saat *offline* sekalipun, peristiwa *realtime* terus berlangsung, sehingga pengguna akhir akan merasakan pengalaman yang responsif. Ketika koneksi perangkat pulih kembali, *Realtime Database* akan menyinkronkan perubahan data lokal dengan *update* jarak jauh yang terjadi selama klien *offline*, sehingga setiap perbedaan akan otomatis digabungkan.

Realtime Database menyediakan bahasa aturan berbasis ekspresi yang fleksibel, atau disebut juga Aturan Keamanan *Firebase Realtime Database*, untuk menentukan metode strukturisasi data dan kapan data dapat dibaca atau ditulis. Ketika diintegrasikan dengan *Firebase Authentication*, *developer* dapat menentukan siapa yang memiliki akses ke data tertentu dan bagaimana mereka dapat mengaksesnya.

Realtime Database adalah *database NoSQL*, sehingga memiliki pengoptimalan dan fungsionalitas yang berbeda dengan *database* terkait. API *Realtime Database* dirancang agar hanya mengizinkan operasi yang dapat dijalankan dengan cepat. Hal ini memungkinkan Anda untuk membangun pengalaman *realtime* yang luar biasa dan dapat melayani jutaan pengguna tanpa mengorbankan kemampuan *respons*. Oleh karena itu, perlu dipikirkan bagaimana pengguna mengakses data, kemudian buat struktur data sesuai dengan kebutuhan tersebut.

Cloud Firestore adalah *database NoSQL* yang *dihosting* di *cloud* dan dapat diakses langsung melalui SDK asli oleh iOS, Android, dan aplikasi *web* Anda. Selain REST (*REpresentational State Transfer*) dan RPC (*Remote Procedure Calls*) API (*Application Programming Interface*), *Cloud Firestore* juga tersedia di *Node.js*, *Java*, *Python*, dan *Go* SDK yang asli.

Setelah model data *NoSQL Cloud Firestore*, simpan data Anda dalam dokumen yang berisi pemetaan kolom terhadap nilai. Dokumen ini disimpan dalam koleksi yang berisi *container* untuk dokumen Anda, yang dapat digunakan untuk mengatur data dan membuat kueri. Dokumen ini mendukung berbagai jenis data, mulai dari *string* dan angka sederhana, hingga objek yang kompleks dan bertingkat. Anda juga dapat membuat

subkoleksi dalam dokumen dan membangun struktur data hierarkis yang berskala sesuai dengan *database*. Model data *Cloud Firestore* mendukung struktur data yang paling sesuai untuk aplikasi Anda.

Selain itu, pembuatan kueri di *Cloud Firestore* bersifat ekspresif, efisien, dan fleksibel. Buatlah kueri dangkal untuk mengambil data pada tingkat dokumen tanpa perlu mengambil keseluruhan koleksi atau subkoleksi bertingkat. Tambahkan pengurutan, penyaringan, dan batasan pada kueri atau cursor untuk memberi nomor pada hasil Anda. Tambahkan *listener realtime* untuk menjaga data di aplikasi Anda tetap terkini, tanpa harus mengambil keseluruhan *database* setiap kali ada *update*. Dengan menambahkan *listener realtime* ke aplikasi, Anda akan mendapatkan pemberitahuan dengan *snapshot* data setiap kali data yang dideteksi oleh aplikasi klien Anda berubah. Dengan begitu, hanya perubahan baru yang akan diambil.

Lindungi akses data Anda di *Cloud Firestore* dengan *Firebase Authentication* dan Aturan Keamanan *Cloud Firestore* untuk Android, iOS, dan *JavaScript*, atau Pengelolaan Akses dan Identitas (IAM) untuk bahasa sisi server.

Developer menggunakan *Firebase* SDK untuk *Cloud Storage* untuk *mengupload* dan *mendownload file* langsung dari klien. Jika koneksi jaringan buruk, klien bisa mencoba operasi ini lagi dari posisi terakhir saat pengoperasian terhenti, sehingga menghemat waktu dan *bandwidth* pengguna.

Cloud Storage menyimpan *file* Anda di *bucket* Google *Cloud Storage*, sehingga membuatnya mudah diakses melalui *Firebase* dan Google *Cloud*. Dengan begitu, Anda memiliki fleksibilitas untuk *mengupload* dan *mendownload file* dari klien seluler melalui *Firebase* SDK, dan melakukan pemrosesan sisi *server* seperti pemfilteran gambar atau transcoding video menggunakan Google *Cloud Platform*. *Cloud Storage* akan diskalakan secara otomatis, yang berarti tidak perlu bermigrasi ke penyedia lain. Pelajari lebih lanjut semua manfaat integrasi dengan *Google Cloud Platform*.

Firebase SDK untuk *Cloud Storage* terintegrasi sempurna dengan *Firebase Authentication* untuk mengidentifikasi pengguna, dan kami

menyediakan bahasa keamanan deklaratif yang dapat Anda gunakan untuk menyetel kontrol akses pada masing-masing *file* atau kumpulan *file*, sehingga Anda bisa menyetelnya sebagai publik atau pribadi sesuai keinginan.

Firebase Cloud Messaging (FCM) adalah solusi pengiriman pesan lintas *platform* yang dapat Anda gunakan untuk mengirim pesan secara tepercaya tanpa biaya. Dengan FCM, Anda dapat memberi tahu aplikasi klien bahwa *email* baru atau data lainnya tersedia untuk disinkronkan. Anda dapat mengirim pesan notifikasi untuk mendorong interaksi kembali dan retensi pengguna. Untuk kasus penggunaan seperti instant *messaging*, pesan dapat mentransfer *payload* hingga 4 KB ke aplikasi klien.

ML Kit adalah SDK seluler yang menghadirkan keahlian *machine learning* Google untuk aplikasi Android dan iOS dalam paket yang andal dan mudah digunakan. Baik masih pemula maupun sudah berpengalaman dalam menggunakan *machine learning*, Anda dapat mengimplementasikan fungsi yang diperlukan hanya dalam beberapa baris kode. Tidak perlu pengetahuan mendalam tentang jaringan *neural* atau pengoptimalan model untuk memulai. Di sisi lain, jika Anda adalah *developer* ML berpengalaman, ML Kit menyediakan API yang mudah digunakan dan dapat membantu Anda menggunakan model *TensorFlow Lite* kustom di aplikasi seluler.

ML Kit mempermudah penerapan teknik ML di aplikasi Anda dengan menghadirkan teknologi ML dari Google, seperti Google *Cloud Vision* API, *TensorFlow Lite*, dan Android *Neural Networks* API secara terpadu dalam satu SDK. Hanya dengan menerapkan beberapa baris kode, ML Kit dapat digunakan untuk memperoleh berbagai hal, seperti pemrosesan berbasis *cloud*, kemampuan *real-time* pada model di perangkat yang dioptimalkan untuk seluler, atau fleksibilitas model *TensorFlow Lite* kustom.

2. Keamanan *Database*

Aturan Keamanan *Firebase* menjadi penghalang antara data Anda dan pengguna yang berniat jahat. Anda dapat menulis aturan sederhana atau kompleks yang melindungi data aplikasi Anda ke tingkat rincian yang dibutuhkan aplikasi spesifik.

Aturan Keamanan *Firebase* memanfaatkan bahasa konfigurasi yang fleksibel dan dapat diperluas untuk mengamankan data Anda di *Cloud Firestore*, *Firebase Realtime Database*, dan *Cloud Storage*. *Firebase Realtime Database Rules* memanfaatkan JSON dalam definisi aturan, sedangkan Aturan Keamanan *Cloud Firestore* dan Aturan Keamanan *Firebase* untuk *Cloud Storage* memanfaatkan bahasa unik yang dibangun untuk mengakomodasi struktur spesifik aturan yang lebih kompleks. Pelajari lebih lanjut tentang cara menyiapkan Aturan untuk produk *Firebase* spesifik yang Anda gunakan di aplikasi Anda, dan bagaimana perilaku Aturan berbeda di seluruh produk *Firebase*.

Aturan Keamanan *Firebase* bekerja dengan mencocokkan pola dengan *path database*, kemudian menerapkan kondisi khusus untuk mengizinkan akses ke data di *path* tersebut. Semua Aturan di seluruh produk *Firebase* memiliki komponen pencocokan *path* dan pernyataan bersyarat yang memungkinkan akses baca atau tulis. Anda harus menetapkan Aturan untuk setiap produk *Firebase* yang Anda gunakan pada aplikasi.

Gambar 2.3 Keamanan *Database*

Aturan diterapkan sebagai pernyataan *OR*, bukan pernyataan *AND*. Akibatnya, jika beberapa aturan cocok dengan jalur, dan kondisi apa pun yang cocok memberikan akses, Aturan memberikan akses ke data di jalur tersebut.

Oleh karena itu, jika aturan luas memberikan akses ke data, Anda tidak dapat membatasi dengan aturan yang lebih spesifik. Namun, Anda dapat menghindari masalah ini dengan memastikan bahwa Aturan Anda tidak terlalu tumpang-tindih. Aturan Keamanan *Firebase* menandai tumpang-tindih di jalur yang cocok sebagai peringatan *compiler*.

Aturan Keamanan *Firebase* juga dapat memanfaatkan *Autentikasi* untuk memberikan izin berbasis pengguna, dan kondisi yang Anda tetapkan bisa sangat mendasar atau sangat kompleks. Pelajari bahasa dan perilaku Aturan lebih lanjut sebelum memulai menulis Aturan.

G. *Quick Respon Code* (QR *Code*)

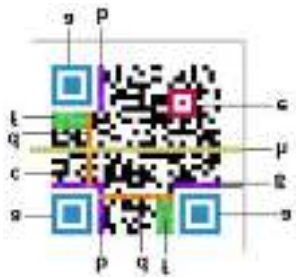
Quick Response Code sering di sebut QR *Code* atau Kode QR adalah semacam simbol dua dimensi yang dikembangkan oleh Denso Wave yang merupakan anak perusahaan dari Toyota sebuah perusahaan Jepang pada tahun 1994. Tujuan dari QR *Code* ini adalah untuk menyampaikan informasi secara cepat dan juga mendapat tanggapan secara cepat. Pada awalnya QR *Code* digunakan untuk pelacakan bagian kendaraan untuk *manufacturing*. Namun sekarang, telah digunakan untuk komersil yang ditujukan pada pengguna telepon seluler.

Gambar 2.4 *QR Code*

QR *Code* biasanya berbentuk persegi putih kecil dengan bentuk geometris hitam (dapat dilihat di gambar 2.1), meskipun sekarang banyak yang telah berwarna dan digunakan sebagai brand produk. Informasi yang dikodekan dalam QR *Code* dapat berupa *URL*, nomor telepon, pesan *SMS*, *V-Card*, atau teks

apapun. QR *Code* telah mendapatkan standarisasi internasional ISO/IEC18004 dan Jepang JIS- X-0510.

1. Anatomi QR *Code*

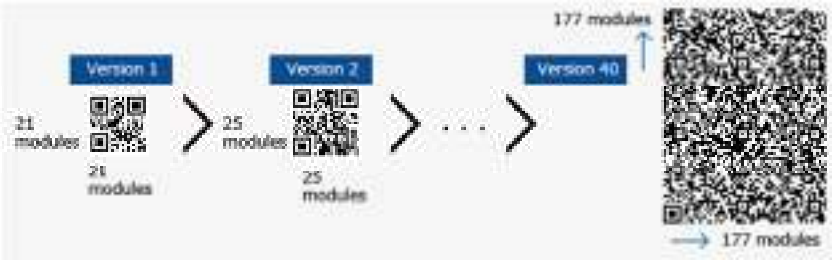


Gambar 2.5 Anatomi QR *Code*

Beberapa penjelasan anatomi QR *Code* antara lain:

a. *Finder Pattern* berfungsi untuk identifikasi letak QR *Code*.
b. Format *Information* berfungsi untuk informasi tentang *error correction* level dan *mask pattern*.
c. Data berfungsi untuk menyimpan data yang dikodekan.
d. *Timing Pattern* merupakan pola yang berfungsi untuk identifikasi koordinat pusat QR *Code*, berbentuk modul hitam putih.
e. Alignment Pattern merupakan pola yang berfungsi memperbaiki penyimpangan QR *Code* terutama *distorsi non linier*.
f. *Version Information* adalah versi dari sebuah QR *Code*.
g. *Quiet Zone* merupakan daerah kosong di bagian terluar QR *Code* yang mempermudah mengenali pengenalan QR oleh sensor CCD
h. *QR Code version* adalah versi dari QR *Code* yang digunakan.

2. Versi QR *Code*



Gambar 2.6 Anatomi Versi *QR Code*

QR *Code* dapat menghasilkan 40 versi yang berbeda dari versi 1 (21 x 21 modul) sampai versi 40 (177 x 177 modul). Tingkatan Versi QR *Code* 1 dan 2 berbeda 4 modul berlaku sampai dengan versi 40. Setiap versi memiliki konfigurasi atau jumlah modul yang berbeda. Modul ini mengacu pada titik hitam dan putih yang membentuk suatu QR *Code*. Setiap versi QR *Code* memiliki kapasitas maksimum data, jenis karakter dan tingkat koreksi kesalahan. Jika Jumlah data yang ditampung banyak maka modul yang yang akan diperlukan dan menjadikan QR *Code* menjadi lebih besar.

H. *Fingerprint* (Sidik Jari)

Fingerprint atau sidik jari adalah hasil reproduksi tapak jari baik yang sengaja diambil, dicapkan dengan tinta, maupun bekas yang ditinggalkan pada benda karena pernah tersentuh kulit telapak tangan atau kaki. Kulit telapak adalah kulit pada bagian telapak tangan mulai dari pangkal pergelangan sampai kesemua ujung jari, dan kulit bagian dari telapak kaki mulai dari tumit sampai ke ujung jari yang mana pada daerah tersebut terdapat garis halus menonjol yang keluar satu sama lain yang dipisahkan oleh celah atau alur yang membentuk struktur tertentu.

Identifikasi sidik jari, dikenal dengan daktiloskopi adalah ilmu yang mempelajari sidik jari untuk keperluan pengenalan kembali identitas orang dengan cara mengamati garis yang terdapat pada guratan garis jari tangan dan telapak kaki. Daktiloskopi berasal dari bahasa Yunani yaitu *dactylos* yang berarti jari jemari atau garis jari, dan *scopein* yang artinya mengamati atau meneliti. Kemudian dari pengertian itu timbul istilah dalam bahasa Inggris, *dactyloscopy* yang kita kenal menjadi ilmu sidik jari.

Fleksibilitas dari gelombang pada kulit berarti tidak ada dua sidik jari atau telapak tangan yang sama persis pada setiap detailnya. Pengenalan sidik jari melibatkan seorang pakar, atau sebuah sistem pakar komputer, yang menentukan apakah dua sidik jari berasal dari jari atau telapak yang sama.

I. *Fused Location Provider*

Fused Location Provider ini merupakan API untuk mendapatkan lokasi dari device android dengan memanfaatkan beberapa sensor dalam perangkat untuk menentukan lokasi perangkat serta dapat mengirim data paling akurat yang tersedia, atau akurasi terbaik yang mungkin tanpa konsumsi daya tambahan.

J. *Representational State Tranfer* (REST)

Representational State Transfer (REST) adalah arsitektur standar *web* yang menggunakan protokol HTTP dalam komunikasi data. Arsitektur tersebut didirikan berdasarkan sumber data dimana masing-masing komponen merupakan sumber data. Sumber data diakses oleh antarmuka yang sama dengan menggunakan metode standar HTTP. Dalam arsitektur REST, *server* yang mengikuti arsitektur REST menyediakan akses ke sumber data dan klien yang mengambil data. Setiap sumber data diidentifikasi menggunakan *link* URI. REST menggunakan berbagai format untuk menyajikan data, seperti teks, JSON dan XML. Berikut adalah metode HTTP yang umumnya digunakan dalam arsitektur REST:

1. *GET* untuk menyediakan akses untuk membaca sumber data.
2. *PUT* untuk memperbarui data yang tersedia.
3. *DELETE* untuk menghapus data.
4. *POST* untuk membuat data baru.

K. *Application Programming Interface* (API)

Application Programming Interface merupakan perangkat definisi subrutin, protokol komunikasi, dan alat untuk membangun perangkat lunak. Spesifikasi API dapat mengambil banyak bentuk, seperti rutinitas, struktur data, kelas objek, variabel, atau pemanggilan jarak jauh.

L. *Unified Modeling Language* (UML)

Menurut Martin Fowler (2005: 1), *Unified Modeling Language* (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendiskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek. Menurut Rosa (2014: 133) UML adalah salah satu standar bahasa yang

banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

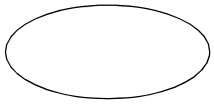
Dari beberapa pengertian di atas, dapat disimpulkan bahwa *Unified Modeling Language* adalah suatu notasi dengan bahasa standar yang digunakan untuk pengembangan berorientasi objek serta menentukan atau menggambarkan suatu sistem *software* yang terkait dengan objek. UML terdiri dari 13 jenis diagram, namun dalam merancang dan membangun aplikasi *Augmented Reality* untuk media pembelajaran pengenalan *hardware* komputer ini, peneliti hanya akan menggunakan 2 macam diagram UML yaitu *use case diagram* dan *activity diagram*.

1. *Use Case Diagram*

   Menurut Martin Fowler (2005: 141), *use case* adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan. Simbol-simbol yang ada pada diagram *use case* ditampilkan pada Tabel 1.1 (Rosa: 2013). Secara sederhana *use case diagram* merupakan gambaran fungsionalitas dari sistem yang dapat diakses oleh *user* atau pengguna.

Tabel 2.1 Simbol *Use Case* Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Segala hal diluar sistem (manusia, sistem, atau perangkat) yang akan menggunakan sistem untuk melakukan sesuatu
2		*Dependency*	Relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain

3		*Generalization*	Menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik
4		*Include*	Menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada digaris dengan panah). *Use case* A dapat berjalan jika *use case* B sudah dijalankan minimal satu kali
5		*Extend*	Menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disisipkan kedalam elemen yang ada di garis dengan panah. *Use case* A memanggil *use case* B pada kondisi tertentu
6		*Association*	Mengidentifikasikan interaksi antara setiap aktor tertentu dengan setiap use case tertentu. Digambarkan sebagai garis antara aktor terhadap *use case* yang bersangkutan
7		*System*	Menyatakan batasan sistem dalam relasi dengan aktor-aktor yang menggunakannya (di luar sistem) dan fiturfitur yang harus disediakan (dalam sistem)
8		*Use Case*	Mengekspresikan tujuan dari sistem yang harus dicapai dan diberi nama sesuai dengan tujuannya

2. *Activity Diagram*

Menurut Martin Fowler (2005: 163), *activity diagram* adalah teknik untuk menggambarkan logika prosedural, proses bisnis dan jalur kerja. Diagram ini memainkan peran mirip sebuah diagram alir yang mendukung behavior paralel. Sedangkan menurut Henderi (2008), *activity diagram* secara grafis digunakan untuk menggambarkan rangkaian aliran aktifitas *use case*.

Tabel 2.2 Simbol *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Start Point*	Menunjukkan dimana aliran kerja dimulai
2		*End Point*	Menunjukkan dimana aliran kerja berakhir
3		*Activities*	Menunjukkan kegiatan dalam aliran kerja
4		*Decision*	Menunjukkan dimana sebuah keputusan perlu diambil dalam aliran kerja
5		*Join*	Menunjukkan percabangan pada aliran kerja
6		*Fork*	Menunjukkan penggabungan dalam aliran kerja

M. *Storyboard*

Menurut Indah Rahmawati (2011: 72) *Storyboard* adalah rangkaian gambar ilustrasi yang berusaha menjelaskan bahasa tulisan *scenario* kedalam bahasa visual. *Storyboard* adalah rancangan berupa *sket* gambar yang dilengkapi dengan petunjuk atau catatan pengambilan gambar untuk kebutuhan *shooting*. Selama proses praproduksi, perancangan yang berhubungan dengan visualisasi yang akan dibuat membutuhkan *Storyboard* sebagai media terpadu. Dilihat dari penjelasan diatas maka dapat disimpulkan *Multimedia audio visual and broadcasting* adalah penggabungan 3 elemen dari unsur-unsur penyampaian ide imajinasi menjadi satu kesatuan untuk menghasilkan sebuah keluaran berupa tampilan gambar dan suara yang menarik.

N. Penelitian Relavan

1. Penelitian Putra dan Wemppy Wantri dari kampus Universitas Sumatra Utara tentang "Sistem Kehadiran Mahasiswa Mengguankan *QR Code* Berbasis

Lokasi dan *Fingerprint* dengan Perangkat Bergerak" pada tahun 2018. Mempunyai kesamaan dengan penelitian saya yang menerapkan penelitian sistem presensi menggunakan Kode QR dan perbedaannya penelitian ini menggunakan *MySQL* sebagai basis datanya.

2. Penelitian Anantassa Fitri Andini, Med Irzal, dan Ria Arafiyah dari kampus Universitas Negeri Jakarta tentang "Perancangan dan Implementasi Sistem Absensi Online Berbasis android di Lingkungan Universitas Negeri Jakarta" mempunyai kesamaan dengan penelitian saya yang menggunakan presensi secara online tetapi bedanya penelitian ini hanya bisa menggunakan IP Wi-Fi di kampus tersebut.
3. Penelitian Benny Yustim, Sriyani Violina dan Adi Purnama dari kampus Universitas Widyatama, Bandung tentang "Perancangan Sistem Monitoring Kehadiran Dosen dan Mahasiswa Berbasis QR Code dan Mobile Technology" pada tahun 2016. Mempunyai kesamaan menggunakan Kode QR untuk presensi.
4. Penelitian Riky Ahmad Fathoni dari kampus Universitas Muhammadiyah Surakarta tentang "Aplikasi Android Pencatatan Kehadiran Kuliah Secara *Real Time* Menggunakan *QR Code* dan *Mobile* Vision di Universitas Muhammadiyah Surakarta" pada tahun 2019. Mempunyai kesamaan dengan penelitian saya yang menerapkan penelitian sistem informasi beserta sistem presensi menggunakan waktu nyata (*Real Time*) dari *Firebase* sebagai data basisnya.
5. Penelitian Norhikmah, Azizah Rahma Safitri dan Laili Annas Sholikhan dari kampus Universitas STMIK AMIKOM, Yogyakarta tentang "Penggunaan QR Code Dalam Kehadiran Berbasis Android" pada tahun 2016. Mempunyai kesamaan dengan penelitian saya yang menerapkan penelitian sistem presensi menggunakan kode QR dan perbedaannya penelitian ini menggunakan *MySQL* sebagai basis datanya.
6. Penelitian Hugo Mulana Christon, Elmor Wagiu dan Yulianus Palopak dari kampus Universitas Advent Indonesia tentang "Perancangan Sistem *E-Learning* Berbasis Android dengan Menggunakan *Firebase* pada Universitas

Advent Indonesia" pada tahun 2018. Mempunyai kesamaan dengan penelitian saya yang menerapkan penelitian menggunakan basis data *Firebase* tetapi penelitian ini mengarahkan belajar online (*e-learning*).

7. Penelitian Muhammad Zaky Faried, Anggraini Mulwinda, dan Yohanes Primadiyono dari kampus Universitas Negeri Semarang tentang "Pengembangan Aplikasi Android Bimbingan Skripsi dengan Fitur Notifikasi" pada tahun 2017. Mempunyai kesamaan dengan penelitian saya yang menerapkan jenis penelitian reserch & development (penelitian & pengembangan), menggunakan database *Firebase* tetapi menggunakan fasilitas *Fire Cloud Messaging* sebagai notifikasi