BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Anil Kumar dkk. (2016) melakukan penelitian tentang membuat sebuah modul berbasis *bluetooth low energy* versi 4 yang berfungsi untuk melacak dan memantau proses kehadiran siswa. Pada penelitian ini dibandingkan antara beacon dengan RFID (*Radio Frequency Identification*), NFC (*Near Field Communication*), dan *Finger Print*. RFID adalah teknologi pengidentifikasi otomatis untuk mendapatkan data tanpa adanya hubungan fisik secara langsung. Namun kelemahannya adalah diperlukan *reader* untuk membaca RFID tersebut. Tentu teknologi ini tidak cocok diimplementasikan untuk pelacakan dan pemantauan, karena akan membutuhkan banyak *reader*, sehingga teknologi tersebut tidak efisien. Selain itu teknologi RFID juga bergantung pada kualitas *tag* atau kartu RFID, *reader*, dan kedekatan antara kartu RFID dengan *reader*.

Selain RFID, dijelaskan juga kekurangan NFC. NFC merupakan *next generation* dari RFID karena mampu menutupi kekurangan RFID. Namun kekurangannya adalah jarak jangkaunya hanya 10 cm dengan kecepatan trasfer data 424 kb/s. Hal tersebut membuat NFC tidak cocok bila dipakai untuk transfer data dalam jumlah yang besar. Lalu yang dibahas selanjutnya adalah tentang *Finger Print*. Presensi dengan menggunakan *Finger Print reader* merupakan teknologi yang hebat. *Reader* menjalankan metode otoritatif dimana dapat mengidentifikasi sidik jari secara spesifika. Namun kekurangan dari *Finger Print reader* ini adalah harganya yang mahal serta rentan terhadap kesalahan pembacaan.

Shota Noguchi dkk. (2015) melakukan perancangan sistem presensi dengan *beacon*. Perancangan tersebut disimulasikan di Universitas Ibaraki, Jepang. Saat itu setiap kartu pelajar mahasiswa telah ditanamkan IC yang berfungsi untuk melakukan presensi. Namun kekurangannya adalah mahasiswa harus melakukan *tapping* atau menempelkan kartu ke *reader* agar dapat melakukan presensi. Tentu

hal tersebut menghabiskan banyak waktu jika semua mahasiswa harus *tapping* satu persatu. Maka dari itu, Shota Noguchi dkk. Membuat sebuah sistem presensi dimana dapat dilakukan secara simultan, yaitu semua siswa dapat melakukan presensi di satu waktu.

Jeon dkk. (2018) melakukan penelitian tentang *Internet of Things* yang berpotensi untuk mengubah gaya hidup manusia agar lebih efektif dan efisien. Salah satu media yang dapat memproyeksikan teknologi IoT tersebut adalah sebuah beacon yang di dalamnya terdapat teknologi Bluetooth *Low Energy*. Adapun fitur yang dapat dilakukan oleh sebuah beacon adalah teknologi *localization, proximity detection*, dan aktivitas penginderaan. Penulis menemukan bahwa *localization* yang diterapkan pada beacon merupakan solusi atas kekurangan yang terdapat pada GPS, Wi-Fi Access point, dan RFID. Pada penelitiannya, dilakukan perbandingan antara beacon BLE (Bluetooth *Low Energy*) dengan Wi-Fi *Access Point*. 19 beacon dan beberapa access point diletakkan pada setiap ruangan di sebuah kantor. Hasilnya Error rate yang dihasilkan beacon adalah sebesar <2.6 m, dengan tingkat akurasi sampai 95%. Sedangkan error rate yang dihasilkan oleh Wi-Fi access point adalah sebesar <8.5 m. Hal tersebut membuktikan bahwa teknologi beacon BLE memiliki tingkat akurasi yang lebih baik dari Wi-Fi Access Point.

Selain *localization*, beacon BLE memiliki tingkat *proximity detection* yang lebih baik dari teknologi lainnya. Keluaran yang dihasilkan oleh deteksi *proximity* adalah hal yang dapat memacu sebuah perintah pada jarak yang telah ditentukan ketika *user* berinteraksi dengan *object*. Adapun teknologi saat ini yang memiliki fungsi tersebut adalah NFC. Namun NFC memiliki kekurangan, yaitu jarak interaksi yang sangat pendek, berkisar antara 10 sampai 20 cm saja, sehingga jika terdapat beberapa *user* di range yang luas, maka proses trigger tidak dapat seluruhnya terlacak oleh semua *user*.

Kekurangan dari penelitian yang dilakukan oleh Jeon dkk. adalah belum dilakukannya implementasi beacon dalam bentuk aplikasi yang merepresentasikan keunggulan dari beacon sebagai alat penunjang Internet of Things. Penelitian

tersebut juga tidak menjelaskan secara rinci tentang seberapa banyak *device* yang dapat di *trigger* oleh sebuah beacon dalam satu waktu.

Okamoto dkk. (2017) melakukan penelitian tentang mengubah metode iklan yang sebelumnya menggunakan pengolahan citra menjadi Bluetooth Low Energy. Mereka menemukan bahwa teknologi citra sebagai cara untuk mengidentifikasi pelanggan pada sebuah toko memiliki beberapa kekurangan, yaitu sulitnya mendapatkan atribut dari pelanggan yang menentukan jenis iklan apa yang akan disampaikan. Pengolahan citra atau *image processing* memiliki tingkat kesalahan yang tinggi karena *image processing* adalah proses untuk mendeteksi, bukan mengidentifikasi dengan pasti. Selain itu, tingkat akurasi pengolahan citra tidak seluruhnya efektif, apalagi jika wajah pelanggan tertutup seperti memakai masker atau syal. Maka dari itu, penulis mengajukan sebuah metode periklanan dengan menggunakan BLE beacons.

Kesimpulan dari penelitian tersebut adalah jumlah atribut yang dapat lebih banyak didapatkan. Penulis melakukan eksperimen terhadap 52 orang, dengan 19 laki-laki dan 33 wanita. Saat dilakukan percobaan untuk mengidentifikasi gender dengan menggunakan *image processing*, hasilnya dideteksi 17 laki-laki dan 27 wanita. Saat memakai beacon, hasil yang didapat adalah ditemukan 18 laki-laki dan 31 wanita. Dari data tersebut didapatkan bahwa beacon dapat mengidentifikasi gender lebih baik daripada *image processing*. Namun kekurangan pada penelitian ini adalah tidak ada penelitian lebih rinci tentang keefektifan penggunaan beacon dalam periklanan. Periklanan dengan beacon pada penelitian ini tidak disimulasikan tentang seberapa sampai iklan tersebut kepada pengguna.

Kaitannya dalam penelitian yang akan diajukan adalah iPresence dapat dijadikan sebuah aplikasi yang mampu memberikan informasi-informasi tentang perkuliahan kepada mahasiswa sesuai dengan atribut atau entitasnya masing-masing, dalam hal ini adalah sesuai dengan mata kuliah yang diambil. Dosen yang merupakan pengampu mata kuliah dapat memberikan informasi kepada mahasiswa, seperti pengunduran jadwal, informasi ujian atau tugas, dan lain-lain. Dosen tidak perlu lagi menghubungi mahasiswa secara personal untuk memberikan

pemberitahuan terkait perkuliahan. Dosen cukup menggunakan aplikasi iPresence untuk menyebarkan pemberitahuan dan informasi akan dikirim langsung kepada mahasiswa.

2.2 Dasar Teori

2.2.1 *Unified Modeling Language*

Unified Modeling Language atau yang biasa disingkat UML merupakan sebuah standar bentuk pemodelan suatu sistem, baik itu *software* maupun konsep pembuatan sistem lainnya. Menurut buku *The Unified Modeling Language Reference Manual Second Edition* yang dikarang oleh Rumbaugh dkk., UML adalah bahasa umum *modeling* visual yang digunakan untuk menentukan, membangun, dan mendokumentasikan *prototype* dari sistem *software* yang akan dibuat. UML menguraikan tentang *design*, *configure*, *maintain*, dan informasi pengendalian sistem. UML dapat digunakan untuk segala metode pengembangan, tahapan *lifecycle*, *application domains*, dan media.

Sebuah standar UML akan merekam informasi tentang struktur *behavior* atau tingkah laku sistem secara statis dan dinamis. Adapun sistem tersebut dimodelkan sebagai kumpulan dari beberapa *class* yang berinteraksi satu sama lain agar program dapat berjalan. Sistem akan dipetakan menjadi dua klasifikasi besar berdasarkan *behavior*, yaitu *static view* dan *dynamic view*.

1. ***Use Case View***

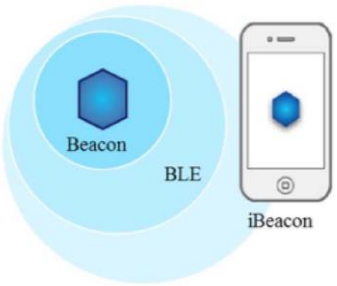
   Use Case view memodelkan fungsionalitas dari sebuah objek yang dirasakan langsung oleh *user* yang disebut dengan *actor*. *Actor* tersebut berinteraksi dengan subjek dari sudut pandang tertentu. *Use case* menjelaskan tentang sebuah transaksi antara *user* dan subjek. Tujuan *use case view* adalah untuk menguraikan seberapa banyak *actor* yang terlibat dalam sistem. Adapun *use case view* digambarkan oleh diagram yang disebut *use case diagram*.

2. ***Activity View***

Activity view merupakan model yang menampilkan diagram alir komputasional pada sistem. Pada model ini akan digambarkan *step by step* dari setiap proses di *background*. Activity view sendiri digambarkan oleh *activity diagram*.

2.2.2 Beacon dan Cubeacon

Beacon merupakan protokol komunikasi yang memungkinkan penyebaran informasi menggunakan Bluetooth *Low Energy* (BLE) versi 4. Menurut Menon dkk. (2017), Beacon merupakan modul dengan penggunaan energi yang lebih efisien serta ekonomis. BLE pada Beacon akan membuat area sinyal regional secara otomatis. Ketika sebuah *device* yang sama-sama memiliki BLE masuk ke area Beacon, maka Beacon akan mengirimkan sinyal kepada *device* tersebut. Beacon dapat diletakkan di permukaan apapun karena



Gambar 3.1 Struktur Sistem Beacon

bentuknya yang *simple* dan kecil. Beacon merupakan sebuah perangkat BLE yang dapat memberikan notifikasi berupa *proximity* yang dianggap tidak mungkin sebelumnya. Beacon dianggap lebih baik dari GPS (*Global Positioning System*) karena tingkat akurasinya lebih akurat sampai 2-100 meter. Beacon juga dapat mendeteksi *device* apapun yang memiliki BLE. Beacon merupakan teknologi yang dinilai lebih baik dari NFC. Jika saja NFC hanya dapat berkomunikasi sejauh 10 cm, maka Beacon memiliki jangkauan hingga 100 meter.

Semua *device* yang memiliki BLE dan diaktifkan akan dianggap satu Beacon sebagai Beacon yang lainnya. Contohnya adalah *device* iPhone yang dijadikan perangkat Beacon. Namun yang membedakan antara BLE pada modul beacon dengan BLE yang terdapat pada *device* iPhone adalah frekuensinya. Singkatnya, Beacon adalah pemancar yang terus menerus menyiarkan sinyak saat sebuah *device* masuk jangkauan Beacon tersebut. Beacon akan mengirimkan serangkaian kode untuk menentukan *action* yang akan dilakukan suatu *device*.

Gambar 3.2 Cubeacon

2.2.3 Android

Android adalah salah satu platform perangkat lunak yang merupakan sistem operasi untuk *mobile device*. Android memiliki basis kernel Linux yang dikembangkan oleh Google dan *Open Handset Alliance*. Android merupakan sistem operasi *open source*, dimana *developer* atau pengembang dapat membuat, mengedit, dan memodifikasi sistem operasi tersebut berbasis bahasa Java. Pada tahun 2005, Google membeli Android. Dikutip dari situs Venture Beat, google membeli Android seharga USD $50 miliar. Keputusan itu terbukti tepat karena saat ini Android sudah menjadi sistem operasi dengan jumlah *user* terbanyak mengalahkan iOS. Sundar Pichai selaku CEO Google menerangkan bahwa sampai bulan Mei 2017, sudah terdapat dua miliar pengguna aktif Android. Selain itu, dikutip dari laman Info Komputer, aplikasi Android pada Play Store sudah menembus angka 2,93 juta aplikasi.

Gambar 3.3 Logo Android

Android selalu dikembangkan oleh pihak Google setiap waktu. Sampai saat ini, Android telah merilis versi terbarunya, yaitu Android versi 8 dengan *codename* Oreo. Berikut adalah tabel versi Android dari waktu ke waktu.

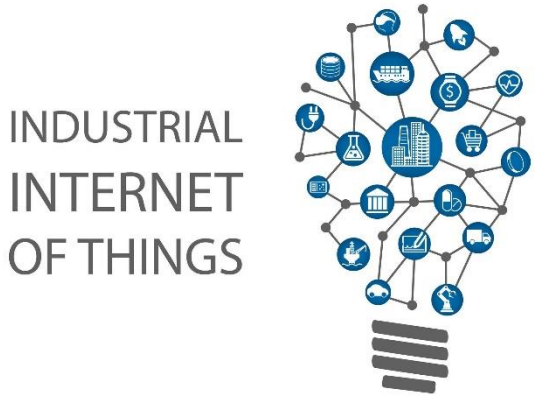
Tabel 2.1 *List* Versi Android

Version	Codename	API
2.3.3 - 2.3.7	Gingerbread	10
4.0.3 - 4.0.4	Ice Cream Sandwich	15
4.1.x	Jelly Bean	16
4.2.x		17
4.3		18
4.4	KitKat	19
5.0	Lollipop	21
5.1		22
6.0	Marshmallow	23
7.0	Nougat	24
7.1		25
8.0	Oreo	26

Saat ini setiap *device* yang terpasang Android memiliki fitur yang lengkap. Untuk konektivitas sudah disediakan beberapa media, yaitu WiFi,

NFC, dan Bluetooth. Bluetooth *Low Energy* sendiri sudah terdapat pada *device* yang memiliki versi Jellybean 4.2. Maka untuk perancangan aplikasi iPresence UMY, digunakan *device* Android yang setara dengan Jellybean 4.2 atau yang lebih baru.

2.2.4 *Internet of Things (IoT)*



Gambar 3.4 Logo Android

Internet of Things atau yang biasa disebut IoT merupakan sebuah paradigma teknologi baru yang mengubah pola hidup manusia. IoT merupakan sesuatu yang terdiri dari miliaran benda fisik dan virtual cerdas yang terintegrasi dengan internet. Menurut Terán dkk. (2017), sebuah sistem dasar yang berbasis IoT mencakup 4 modul utama, yaitu:

1. *Sensing* (merekam variabel fisik)
2. *Embedded systems* (pengenalan dan pemrosesan)
3. *Data Communication* (konektivitas)
4. *Data Analytic* (menemukan gagasan)

IoT merupakan satu atau lebih *device* yang terhubung internet yang dapat dikendalikan secara jarak jauh. Pada pengimplementasiannya, IoT membutuhkan banyak *platform* dan teknologi yang berbeda, karena hal tersebut berkaitan dengan proses identifikasi dan pelacakan, komunikasi, komputasi, pengontrolan, pemrosesan, termasuk lalu lintas data. Tentu semua

itu tidak bisa dijalankan hanya dengan satu alat, tetapi membutuhkan banyak alat sesuai dengan fungsinya.

2.2.5 Java *Code*

Gambar 3.5 Logo Java

Java merupakan bahasa pemrograman yang dapat dijalankan di berbagai *device*, mulai dari komputer hingga *handphone*. Java diciptakan oleh James Gosling dari perusahaan Sun Microsystem pada tahun 1991. Versi pertama Java yaitu Java 1.0 yang dirilis pada tahun 1995. Dari waktu ke waktu pemrograman Java terus dikembangkan dan ditambahkan banyak *libraries*. Sampai sekarang versi terbaru dari Java adalah Java versi 8 yang baru dirilis pada tanggal 17 Oktober 2017. Java sendiri banyak mengadopsi sintaksis dari bahasa pemrograman C dan C++, lalu kemudian dibuat lebih sederhana.

Saat ini pemrograman Java telah digunakan oleh banyak *platform*. Java sendiri menjadi pondasi bagi bahasa pemrograman lainnya, seperti Kotlin, Scala, Clojure, Groovy, Jruby, Jython, dan lainnya. Semua pemrograman tersebut memanfaatkan Java *Virtual Machine* sebagai rumahnya. Selain itu, Java digunakan untuk pembuatan aplikasi *native* untuk Android.

Pemilihan bahasa Java sebagai bahasa pemrograman pada android memiliki alasan. Dikutip dari laman web Bahasa Java, terdapat beberapa

alasan Google menjadikan Java sebagai bahasa pemrograman pada android. Berikut penjelasannya:

1. Java merupakan bahasa pemrograman yang populer. Setiap pengembang tidak perlu lagi mempelajari bahasa pemrograman yang baru, dikarenakan Java sudah familiar dan merupakan bahasa paling terkenal di dunia.
2. Dikarenakan Java terkenal, maka pengembang atau *developer* dari pemrograman java sudah banyak. Dengan banyaknya pengembang java, maka *tools* yang tersedia pun banyak. Banyak *libraries* yang tersedia, sehingga *developer* lain lebih dimudahkan karena tidak perlu lagi membuat suatu *method* baru.
3. Java beroperasi dalam *virtual machine*, sehingga tidak perlu melakukan proses *recompile*. Proses pemisahan operasi dari aplikasi satu sama lain lebih mudah karena antara satu aplikasi dengan aplikasi lainnya tidak akan saling mengganggu.

2.2.6 Android Studio

Untuk mengembangkan sebuah aplikasi, maka diperlukan sebuah *tools*. Aplikasi tidak bisa terbentuk hanya dengan satu *tool***,** maka dari itu terdapat suatu istilah yang menggabungkan semua *tools* pengembangan aplikasi menjadi satu, yaitu IDE (*Integrated Development Environment*). IDE setidaknya harus memiliki 4 fasilitas utama, yaitu:

1. *Editor*, yang berfungsi untuk menuliskan *source code* dari perangkat lunak.
2. *Compiler*, yaitu fasilitas untuk melakukan pengecekan terhadap sintaks dan *source code*, lalu kemudian menerjemahkannya menjadi bahasa mesin (*binary*).
3. *Linker*, yaitu berfungsi untuk menyatukan data-data *binary* menjadi suatu program yang siap dieksekusi.

4. *Debugger*, yaitu berfungsi untuk melakukan pengetesan jalannya program tersebut. *Debugger* memiliki fungsi sebagai pencari *bugs* atau kesalahan yang terdapat pada program.

Saat ini terdapat beberapa IDE untuk pengembangan aplikasi android. Salah satunya adalah Android Studio. Android Studio merupakan IDE yang resmi dirilis pada tanggal 16 Mei 2013 di Google I/O *conference*. Android

Gambar 3.6 Logo Android Studio

Studio merupakan perangkat lunak gratis yang lisensinya dimiliki oleh Apache Lisence 2.0. Pertama, Google merilis Android Studio dalam versi *preview* versi 0.1. Setelah versi *preview*, Google mengeluarkan Android Studio versi beta 0.8 pada bulan Juni 2014. Versi *stable* dari Android Studio baru dirilis pada bulan Desember 2014, dimulai pada versi 1.0. Adapun versi terbaru dari Android Studio sekarang adalah versi 3.0.0.

Selain Android Studio, terdapat beberapa IDE *tools* lainnya, yaitu Eclipse, Xamarin, App Inventor, React Native, Ionic, dan yang lainnya. Namun penulis menggunakan Android Studio karena beberapa kelebihan. Adapun kelebihan Android Studio dari IDE *tools* lainnya adalah sebagai berikut:

1. ***Intelligent Code Editor***

   Intelligent Code Editor merupakan suatu fitur khusus yang menjadi kelebihan paling utama dari Android Studio. Dengan fitur tersebut, Android Studio memiliki kemampuan *auto completion*, yaitu menampilkan *suggestion* untuk sebuah *syntax* yang *error*. Android Studio mampu menganalisis sebuah kode

dengan mumpuni serta dapat melakukan *refactoring*, sehingga waktu pembuatan program menjadi lebih cepat dan lebih produktif.

2. ***Instant Run***

   Fitur *Instant Run* merupakan fitur yang berfungsi untuk mempercepat *compile* dan *run* saat dilakukan untuk kedua kali. *Compile* dan *run* untuk pertama kalinya tentu lama. Namun setelah itu, saat program di *compile* dan *run* lagi, maka waktunya jauh lebih cepat. Hal itu dikarenakan Android Studio tidak membuat ulang APK, melainkan hanya mengaplikasikan perubahan dari hasil file apk sebelumnya.

3. **Sistem *build* yang Handal dan Fleksibel**

   Fitur ini akan memudahkan pengembang saat melakukan *compile*. Saat perintah *compile* dieksekusi, maka secara otomatis file apk akan terbentuk, Sehingga pengembang tidak perlu lagi *build* aplikasi kembali.

2.2.7 Adobe Photoshop CC

Interface merupakan bagian yang harus ada dalam sebuah aplikasi. Interface berfungsi sebagai jembatan antara *back-end service* dan *user*. Maka dari itu, sebuah aplikasi memerlukan *interface*. Adobe Photoshop CC berfungsi untuk membuat *design interface* berupa *mock up* dari sebuah aplikasi. *Mock up* sendiri adalah sebuah *design* yang merupakan gambaran tentang bagaimana tampilan dan jalannya sebuah program. *Design* yang dibuat adalah semua tampilan, dari awal aplikasi, hingga cara menutup aplikasi. Adobe Photoshop CC memiliki sebuah *template* yang disebut dengan *artboard* yang merupakan lembaran untuk membuat *design mock up*.

2.2.8 Mesosfer

Gambar 3.7 Logo Mesosfer

Mesosfer merupakan sebuah *mobile back end as a service* (MBaaS) yang berperan untuk menyatukan fungsi sebuah *mobile app* dengan *Internet of Things* (IoT). Mesosfer berfungsi sebagai pengendali *server cloud*, integrasi protokol, komunikasi, *storage*, dan oAuth (protokol internet). Dengan kata lain, Mesosfer berperan penting untuk mengendalikan, mengelola, dan mengirimkan perintah ke perangkat IoT.

Aplikasi iPresence menggunakan Mesosfer sebagai MBaaS sekaligus *database* semua data, yaitu data identitas *user*, data kuliah yang meliputi ruang kuliah dan mata kuliah, dan data presensi. Mesosfer sudah dilengkapi fitur Mesosfer SDK yang berfungsi sebagai basis pengembangan aplikasi yang terintegrasi dengan Mesosfer. Mesosfer SDK mencakup *library-library* untuk memudahkan *developer* untuk mengoperasikan *database* via Mesosfer melalui *mobile app*.