BAB II

LANDASAN TEORI

2.1. *Context-aware Ubiquitous Learning*

Context-aware Ubiquitous Learning merupakan konsep pembelajaran yang memperbolehkan mahasiswa untuk belajar dengan menggunakan beragam jenis gawai *mobile* sebagai fasilitator lingkungan pembelajaran yang mulus dan selalu tersedia. Konsep ini mementingkan prinsip pembelajaran pada waktu yang tepat, tempat yang tepat, sumber daya yang tepat, dengan cara yang tepat. Dalam mencapai lingkungan yang sifatnya context-aware dan seamless, beberapa teknologi ubiquitous computing biasa digunakan untuk mendeteksi konteks informasi yang dibutuhkan oleh pengguna yaitu: RFID, GPS, sensor, kartu pintar, komputer *wearable,* dan protokol-protokol komunikasi nirkabel. Informasi yang diperoleh tadi tidak hanya digunakan untuk melakukan seleksi terhadap kondisi pengguna, melainkan untuk dimanfaatkan sebagai pendukung pedoman pembelajaran. Melalui integrasi secara fisik, mahasiswa juga dapat memahami benda-benda fisik secara nyata dan melakukan aktivitas tertentu secara *learner-centered*, sesuai dengan ilmu pendidikan yang kondusif (Jeng, Wu, Huang, Tan, & Yang, 2010).

2.1.1. *Electronic Learning*

Electronic learning merupakan kumpulan dari berbagai disiplin yang berbeda-beda (Sangrà, Vlachopoulos, & Cabrera, 2012). Dari sudut pandang teknologi, *electronic learning* adalah penggunaan media elektronik untuk berbagai tujuan pembelajaran. Tujuan-tujuan tersebut meliputi fungsi tambahan pada kelas konvensional sampai penggantian *electronic learning* sebagai kelas *online* (Guri-Rosenblit, 2005). Sedang dari orientasi sistem penyampaian data, *electronic learning* merupakan penyampaian pendidikan itu sendiri melalui berbagai media elektronik (Koohang & Harman, 2005). Dari sudut pandang aspek komunikasi, *electronic learning* merupakan pembelajaran berbasis teknologi informasi dan komunikasi dengan interaksi pedagogis antara: mahasiswa dan konten; mahasiswa dan dosen; atau antar mahasiswa melalui internet (Videgaray, 2007). Terakhir, berkaitan dengan paradigma pendidikan, *electronic learning* mengacu pada proses edukasi yang memanfaatkan teknologi informasi dan komunikasi sebagai mediator pembelajaran secara langsung maupun tidak langsung (Jereb & Šmitek, 2006).

2.1.2. *Mobile Learning*

Mobile learning mengacu pada penggunaan gawai *mobile* atau nirkabel untuk tujuan belajar dimana pun. Contoh gawai yang dapat digunakan meliputi ponsel pintar, komputer genggam, komputer tablet, komputer jinjing, dan lain-lain (Traxler & Kukulska-Hulme, 2005). *Mobile learning*

memungkinkan mahasiswa untuk berinteraksi menggunakan fitur tambahan seperti pesan teks, akses internet *mobile*, dan komunikasi suara melalui jaringan nirkabel. Menurut Koole (2009), terdapat beberapa kelebihan dari *mobile learning*:

i. Gawai *mobile* yang tersambung pada jaringan nirkabel memungkinkan mahasiswa dapat mengakses informasi relevan dimanapun dan kapanpun saat dibutuhkan.

ii. Kemampuan untuk mengakses berbagai macam materi untuk meningkatkan pemahaman maupun daya ingat tanpa terbatas tempat dan waktu.

iii. Belajar dengan konteks tertentu dapat membantu menghubungkan budaya maupun lingkungan setempat dengan pemahaman akan kegunaan dari informasi tersebut. Lagi-lagi untuk membantu mengingat dan memanggil kembali ingatan tersebut dengan lebih mudah.

iv. Edukasi *mobile* yang diimplementasikan dengan baik dapat membantu mengurangi beban kognitif dari mahasiswa itu sendiri. Meskipun memang susah untuk mengetahui betul bagaimana cara membagi-bagi informasi untuk diingat, pada akhirnya ragam informasi kontekstual mungkin bisa membantu mahasiswa untuk menahan, mengingat, maupun mengekspresikan informasi tadi saat dibutuhkan.

2.1.3. Ubiquitous Learning

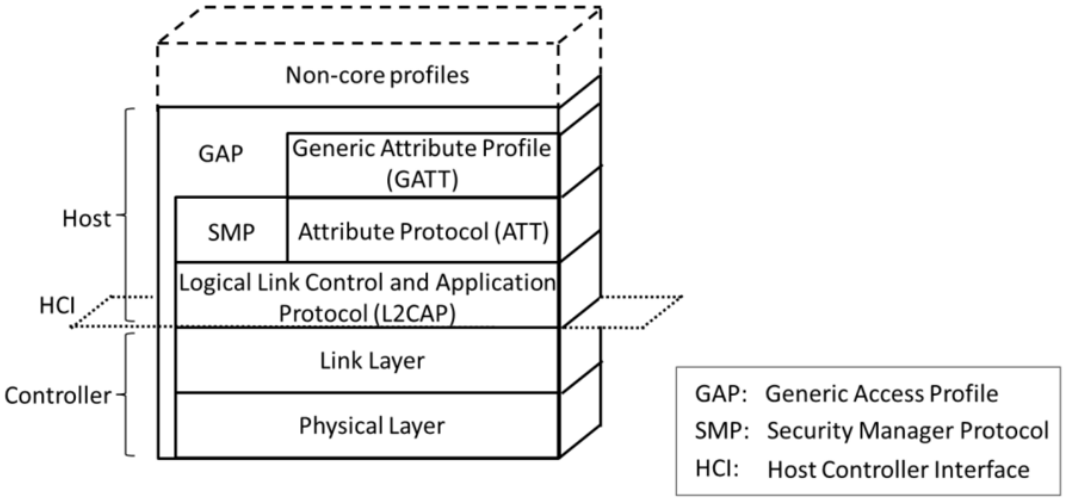
Menurut Cope & Kalantzis (2009), *ubiquitous learning* adalah sebuah paradigma pendidikan baru yang dimungkinkan oleh adanya keterjangkauan media *digital*. Terdapat tujuh gerakan yang menjadi karakteristik dari *ubiquitous learning*:

I. Untuk mengaburkan batasan-batasan kelembagaan, tempat, dan waktu dari edukasi tradisional.
II. Untuk menggeser keseimbangan agensi.
III. Untuk mengetahui perbedaan mahasiswa dan memanfaatkan informasi tersebut sebagai sumber daya produktif.
IV. Untuk memperluas jangkauan dan ragam representasi edukasi.
V. Untuk mengembangkan kemampuan konseptual.
VI. Untuk menghubungkan pikiran satu individu ke dalam pengetahuan kolektif.
VII. Untuk membangun budaya pengetahuan yang kolaboratif.

2.2. *Bluetooth Low Energy* (BLE)

Menurut Gomez, Oller, & Paradells (2012), Bluetooth Low Energy (BLE) adalah teknologi nirkabel yang dikembangkan oleh Bluetooth Special Interest Group (SIG) untuk komunikasi jarak dekat, yang lebih dikhususkan penggunaannya sebagai solusi hemat energi dalam aplikasi *control and monitoring*. Solusi BLE untuk kehidupan sehari-hari dapat mencakup *use cases* pada sektor kesehatan, elektronik konsumen, energi pintar, dan keamanan.

Seperti teknologi *bluetooth* pada umumnya, susunan protokol dari BLE terdiri dari 2 bagian utama: *controller* dan *host*. Bagian *controller* mencakup *physical layer* dan *link layer*, dimana biasanya bagian-bagian tersebut dirakit sebagai sebuah *System-on-Chip* (SOC) yang terintegrasi dengan piranti radio. Sedang, bagian *host* berjalan pada *application processor* dan memuat fungsi-fungsi *upper layer (L2CAP, ATT, GATT, SMP, GAP)*. Komunikasi antara bagian host dan controller distandarisasikan sebagai Host Controller Interface (HCI). Pada akhirnya, *application layer* dapat dijalankan di atas *host* untuk memenuhi kebutuhan *user*.



Gambar 2. 1. Lapisan Protokol BLE.

Sumber : (Gomez, Oller, & Paradells, 2012)

2.2.1. *Bluetooth Beacon*

Bluetooth beacon merupakan sebuah piranti keras yang memiliki fungsi untuk mengirimkan potongan data dalam jarak dekat menggunakan protokol

Bluetooth Low Energy. Potongan data tersebut dapat berisi informasi berupa teks singkat yang bisa memberikan URL (*Uniform Resource Locator*) ataupun informasi yang bersangkutan lainnya. Ponsel pintar yang telah mendukung protokol BLE dapat menerima sinyal data tadi untuk ditampilkan kepada penggunanya. Interaksi tersebut menghasilkan pemahaman *user* akan informasi kontekstual mengenai entitas yang direpresentasikan dalam jarak deteksi *beacon* tersebut (Bhattacharya, Canul, & Knight, 2017)

2.2.2. Eddystone *Format*

Eddystone merupakan sebuah *format beacon* terbuka buatan Google yang menjunjung transparansi dan keamanan. *Beacon* yang menggunakan format Eddystone dapat dideteksi oleh gawai berbasis Android maupun iOS. Format Eddystone dibangun dari berbagai ilmu yang telah didapatkan oleh gabungan rekanan industri dengan Google saat melakukan penelitian dan pembangunan dalam teknologi *beacon*, juga dengan komunitas *beacon* yang lebih luas lagi.

Terdapat beberapa varian paket yang dapat disisipkan dalam *format frame* Eddystone, yang meliputi:

1. Eddystone-UID: Sebuah nilai *identifier* yang tidak berubah dan unik, disusun oleh komponen *namespace* sebesar 10 *byte* dan komponen *instance* sebesar 6 *byte*.

2. Eddystone-URL: Sebuah alamat URL terkompresi, yang dapat digunakan langsung oleh klien setelah dilakukan *parsing* dan dekompresi.
3. Eddystone-TLM: Merupakan data-data yang mencakup status dari *beacon* itu sendiri. Berguna untuk melakukan perawatan berkala dan menjadi fondasi bekerjanya sistem pemeriksaan menggunakan Google Proximity Beacon API.
4. Eddystone-EID: Sebuah frame yang dapat berubah-ubah sesuai waktu (*time-varying*), dimana dapat dimanfaatkan sebagai *identifier* bagi aplikasi *resolver*, seperti Proximity Beacon API.

Selain keempat frame tadi, Eddystone juga dapat menyajikan aplikasi untuk melakukan konfigurasi maupun mendukung materi branding (Google, 2018).

2.2.3. Estimote *Beacon*

Estimote *Beacon* merupakan piranti keras *beacon* yang memiliki fitur *bi-directional* BLE *radio* yang dijual bebas untuk konsumsi khalayak umum (Estimote, Inc., 2018). Piranti tersebut dapat dipasang pada objek apapun di lokasi manapun, yang kemudian akan mengirim sinyal radio BLE yang dapat diterima dan dimengerti oleh ponsel pintar. Sinyal tersebut kemudian dapat memberikan informasi terhadap pengguna ponsel mengenai area di sekitarnya ataupun objek yang berada di dekatnya. Dibutuhkan gawai dengan standar Bluetooth 4.0 atau lebih tinggi untuk dapat mengakses sinyal dari Estimote Beacon.

Piranti ini memiliki *chip* nRF51822, sebuah *System-on-Chip* (SOC) multiprotokol yang kuat dan fleksibel. *Chip* tersebut dirakit menggunakan prosesor 32-bit ARM Cortex M0 CPU dengan *flash* sebesar 256/128kB beserta RAM sebesar 32/16kB. Seluruh SOC ini dirancang untuk menjadi sistem yang hemat energi dan efisien. Terdapat beberapa jenis data yang dikirim dalam satu paket sinyal, yaitu:

1. Alamat MAC
2. *Universally Unique Identifier* (UUID)
3. Nomor *major*, yang digunakan untuk membagi *sets* menjadi *segments*.
4. Nomor *minor*, yang digunakan untuk membagi *segments* menjadi *subsegments*.

(Kriz, Maly, & Kozel, 2016)

2.3. Sistem Operasi Android

Android merupakan sebuah susunan piranti lunak *open source* yang diciptakan untuk berbagai jenis gawai dengan rupa bentuk yang beragam pula. Tujuan utama dari sistem operasi Android adalah untuk menciptakan sebuah platform piranti lunak terbuka yang tersedia bagi berbagai operator jaringan, manufaktur gawai, dan pengembang piranti lunak untuk merealisasikan ide-ide inovatif menjadi produk konkret yang meningkatkan pengalaman *mobile* bagi para penggunanya (Google, 2018).

2.3.1. Android Software Stack

Menurut Google (2018) dan Techotopia (2016), terdapat beberapa komponen penyusun arsitektur *platform* Android untuk menjalankan fungsi sistem operasi yang disebut Android Software Stack, yakni:

1. Linux *Kernel*

   Linux *Kernel* menyediakan sebuah lapisan pemisahan antara piranti keras gawai dan lapisan-lapisan atas dari Android *software stack*. *Kernel* yang berbasis pada Linux versi 2.6 menyediakan fitur *multitasking, low-level core system services* seperti pengaturan memori, proses, dan manajemen daya. Selain itu *kernel* ini juga menyediakan sebuah lapisan jaringan dan *driver* piranti keras seperti *display*, Wi-Fi, maupun audio.

2. *Hardware Abstraction Layer* (HAL)

   Hardware Abstraction Layer (HAL) menyediakan antarmuka standar yang menghubungkan piranti keras gawai dan bahasa tingkat tinggi Java API *Framework*. HAL terdiri dari beberapa modul *library*, dimana masing-masing mengatur antarmuka dari berbagai jenis komponen seperti kamera dan modul *bluetooth*. Ketika *framework* API memanggil akses pada piranti keras gawai, sistem Android akan menjalankan modul *library* yang bersangkutan dengan piranti keras gawai tersebut.

3. Android *Runtime*

Untuk gawai yang menggunakan Android versi 5.0 (API level 21) atau lebih baru, masing-masing aplikasi berjalan di atas prosesnya sendiri bersamaan dengan versi Android Runtime (ART)-nya sendiri. ART dibuat untuk menjalankan banyak *virtual machine* pada gawai yang memiliki memori kecil dengan cara menjalankan *file* DEX. *File* DEX merupakan sebuah format *bytecode* yang didesain khusus untuk penggunaan memori sesedikit mungkin (50% lebih rendah daripada *bytecode* Java pada umumnya).

4. *Native* C/C++ *Libraries*

Mayoritas komponen maupun *services* pada sistem operasi Android seperti ART dan HAL dibangun dari *native code* yang membutuhkan *native libraries* dengan bahasa C dan C++. Platform Android menyediakan Java Framework API untuk memberikan fungsionalitas *native libraries* tadi pada berbagai aplikasi di sistem Android. Misalnya, pengembang aplikasi dapat mengakses fitur OpenGL ES melalui *framework* Java OpenGL API untuk memberikan dukungan terhadap penampilan grafis dua dimensi dan tiga dimensi pada aplikasi yang dikembangkannya.

5. Java API *Framework*

Java API *Framework* merupakan kumpulan dari berbagai bentuk *application programming interfaces* (APIs) untuk membangun beragam

aplikasi Android yang didasarkan pada konsep modularitas. Contoh beberapa *framework* dan kegunaan nya, sebagai berikut:

I. View System

Digunakan untuk membangun *user interface* dari sebuah aplikasi secara ekstensif. Terdapat fitur-fitur membantu *seperti list, grid, text box, button,* sampai *web browser embeddable* yang dapat dimanfaatkan dalam mencapai *user experience* terbaik pada pengembangan aplikasi.

II. Resource Manager

Menyediakan akses pada sumber daya bukan kode seperti *localized string*, aset grafis, maupun aset *layout*.

III. *Notification Manager*,

Memungkinkan aplikasi apapun untuk menampilkan informasi maupun peringatan lain pada *status bar*.

IV. *Activity Manager*,

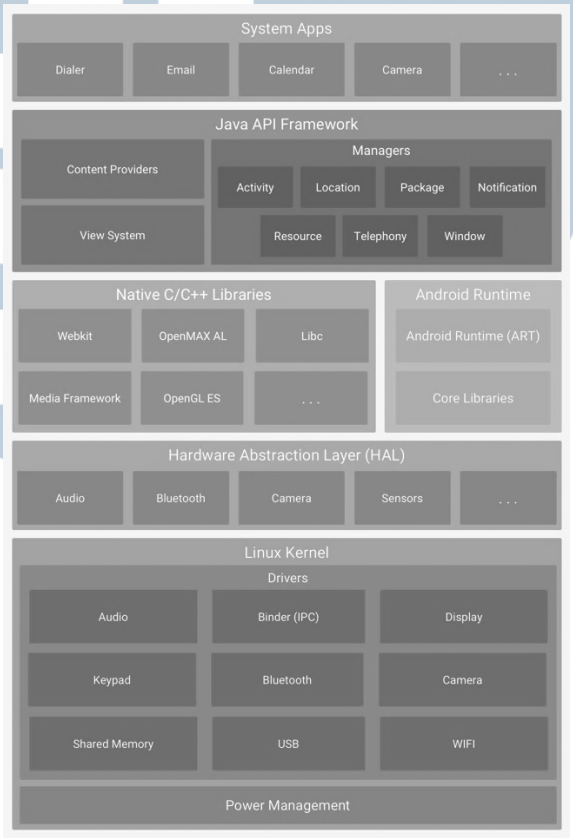
Mengatur siklus kehidupan seluruh aplikasi dalam sebuah sistem Android dan menyediakan sebuah sistem navigasi *back stack* kolektif.

V. *Content Provider*,

Memungkinkan berbagai aplikasi untuk mengakses data dari aplikasi lainnya, seperti menarik data teman dari aplikasi kontak untuk kebutuhan tertentu.

6. *System Application*

System Application merupakan lapisan teratas pada susunan sistem operasi Android. Lapisan ini terdiri dari aplikasi *native* atau bawaan yang tersedia saat gawai berbasis Android pertama kali digunakan dan aplikasi tambahan pihak ketiga yang dipasang oleh pengguna setelah mendapatkan gawai Android.



Gambar 2. 2. Android Software Stack

Sumber : (Google, 2018)

2.3.2. MySQL *Database*

Menurut Oracle (2018), MySQL merupakan sebuah sistem manajemen basis data relasional yang bersifat *open-source*. Aplikasi dari sistem basis data MySQL sendiri merupakan sistem *client-server* yang terdiri dari: SQL

Server *multi-thread* yang mendukung berbagai macam aplikasi *back-end*; berbagai varian aplikasi dan *library* klien; aplikasi administratif; dan berbagai macam *application programming interfaces* (APIs). Selain itu, MySQL Server juga bisa dijadikan sistem *standalone*, dimana tidak memerlukan koneksi *client-server* sehingga *user* dapat meraih sistem basis data yang lebih ringkas, cepat, dan mudah untuk diatur.

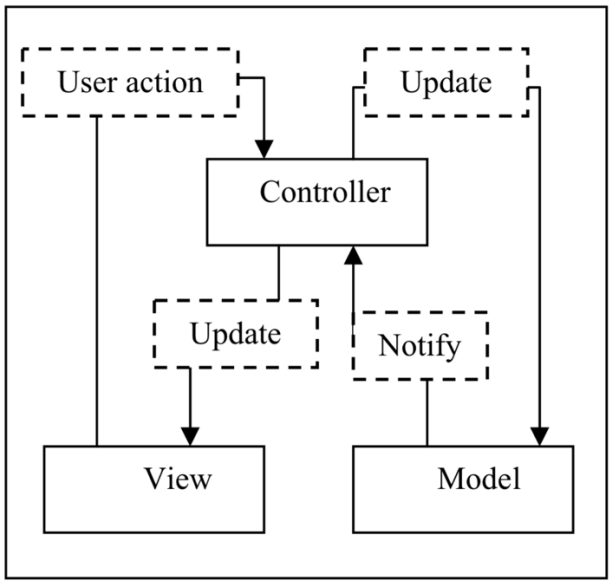
2.4. Pemrograman Web *Model-Visual-Controller* (MVC)

Menurut Wang (2011), Arsitektur *Model-Visual-Controller* (MVC) merupakan konsep pemrograman web yang menggunakan pola pemisahan antara logika domain aplikasi dan antarmuka grafis aplikasi tersebut sehingga pengembangan, pengetesan, dan perawatan aplikasi web dapat dilaksanakan secara independen. Berikut merupakan struktur dari arsitektur MVC:

i. *Model* mengelola tindakan dan data dari domain aplikasi. Terdapat dua jenis respon yang dilakukan: memberi informasi mengenai *state* yang diminta dan merubah *state* sesuai dengan instruksi. Dalam sebuah sistem *event-driven*, model memberitahu *observers* saat informasi berubah supaya mereka dapat bereaksi.

ii. *View* menampilkan model menjadi bentuk yang sesuai untuk interaksi dengan pengguna, biasanya dalam rupa elemen antarmuka grafis. Satu model dapat memiliki beberapa *views* untuk tujuan tertentu. Sebuah *viewport* biasanya memiliki korespondensi *one to one* dengan permukaan layar dan memiliki protokol untuk menampilkannya.

iii. *Controller* menerima masukan dan menginisiasi respon dengan melakukan panggilan terhadap objek model. Sebuah controller menerima masukan dari pengguna lalu menginstruksikan model dan *viewport* untuk melakukan aksi tertentu berdasar masukan tersebut.



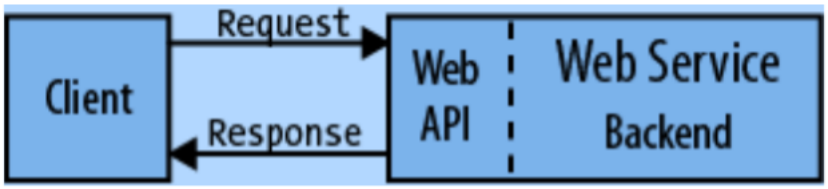
Gambar 2. 3. Mekanisme MVC

Sumber : (Wang, 2011)

2.5. *Representational State Transfer Application Programming Interface* (REST API)

Menurut Massé (2012), *Representational State Transfer Application Programming Interface* merupakan sebutan untuk layanan penyedia kumpulan data dan fungsi-fungsi untuk memfasilitasi interaksi antara *web service* dan perangkat klien seperti *website* atau jenis aplikasi apapun. Selain sebagai penyedia, terdapat

satu fungsi dimana dalam perancangannya mengikuti panduan arsitektur *Representational State Transfer*. Sehingga, dapat dikatakan bahwa sebuah layanan *web* yang telah mengikuti arsitektur *Representational State Transfer* tersebut adalah sebuah *web* yang memiliki desain '*RESTful'*.



Gambar 2. 4. Arsitektur *Representational State Transfer*

Sumber : (Ong, et al., 2015)

Dalam sebuah desain *'RESTful'*, setiap objek direpresentasikan sebagai sumber daya yang unik, dan dapat diminta menggunakan struktur *queries* yang seragam. Cara penggunaan layanan web yang telah mengusung desain tersebut mengikuti metode-metode yang dimiliki *hypertext transfer protocol* (HTTP) seperti *GET, POST, PUT*, dan *DELETE* bersamaan dengan alamat *uniform resource identifier* yang unik pula (Ong, et al., 2015).

2.6. Agile Unified Process

Menurut Edeki (2013), siklus pengembangan sistem *Agile Unified Process* merupakan konsep yang berdasar pada metode *agile* dan *Rational Unified Process* rancangan Scott W. Ambler, yang memberikan pendekatan iteratif-*incremental* pada pengembangan piranti lunak. *Agile Unified Process* terdiri dari tujuh alir kerja, dimana masing-masingnya memiliki empat fase. Alir kerja tersebut meliputi *model, implementation, test, deployment, configuration management, project*

management, dan *environment*. Berikut merupakan fase-fase yang terdapat pada siklus pengembangan ini:

1. Fase *Inception*, yang dimulai dari alir kerja model dimana lingkup proyek, resiko, biaya, penjadwalan, dan kelaikan didefinisikan.
2. Fase *Elaboration*, dimana persyaratan diterjemahkan menjadi diagram-diagram *unified modelling language*, desain antarmuka grafis, dan rancangan model lainnya yang mendukung fase *construction*.
3. Fase *Construction*, yang merupakan fase terbesar dan kebanyakan menitikberatkan pada penciptaan kode-kode yang dapat dieksekusi.
4. Fase *Transition*, dimana hasil peranti lunak diserahkan pada pengguna. Karena sifat iteratif-*incremental* dari siklus ini, maka setelah fase transition pengembang dapat kembali ke fase *inception*, *elaboration*, *construction*, maupun *transition* lagi untuk pengembangan selanjutnya.

2.7. Penelitian Terdahulu

Tabel 2. 1. Penelitian Terdahulu

No.	Nama Peneliti	Judul Artikel	Tahun	Hasil yang dipakai
1.	Debasis Bhattacharya, JD, DBA., Mario Canul, dan Saxon Knight	"*Case Study: Impact of the Physical Web and BLE Beacons*" diambil dari "*Proceedings of the 50th Hawaii International Conference on System Sciences"*	2017	Penggunaan beacon pada lingkup akademis kampus layak untuk diimplementasi; Penggunaan Eddystone Ephemeral ID (EID) dapat dimanfaatkan untuk memberikan akses deteksi beacon hanya pada user yang sudah di registrasi.
2.	G. Tabunshchyk, Dirk Van Merode, Yriy Goncharov, dan Konstantin Patrakhalko	"*Smart-Campus Infrastructure Development Based On BLE 4.0*" diambil dari "*Journal Electrotechnic and Computer Systems No. 18 (94)"*	2015	Penggabungan sistem manajemen konten atau CMS terhadap sistem basis data yang terhubung dengan *trigger beacon* dapat diimplementasikan.
3.	Samarth R. Gohel, Ashvini M. Padavi, Mohit M. Asija, dan Purnanand A. Vasave	"*Bluetooth beacon based Attendance System with Android Application*" diambil dari "*Journal of Android and IOS Applications and Testing Volume 3 Issue 1"*	2018	Pada sistem absensi berbasis teknologi *bluetooth beacon* disimpulkan bahwa implementasi sistem dapat menghemat waktu dan sumber daya.