BAB II
TINJUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Token adalah kode pengaman untuk aplikasi presensi mahasiswa. *Token* ini berfungsi untuk mengeluarkan *dynamic password* (kode dinamis), yaitu kode yang selalu berubah dan hanya dapat digunakan satu kali untuk tiap kali presensi yang dilakukan.

Menurut Kalaikavitha (2013), *One time password* (OTP) merupakan teknik autentikasi yang hanya berlaku untuk satu kali penggunaan OTP menjamin keamanan terhadap serangan *replay*, dimana penyerang yang berhasil mendapatkan nilai OTP yang sudah digunakan tidak mungkin dapat digunakan kembali. OTP menggunakan algoritma pembangkitan dengan sifat semu acak yang telah diuji secara kriptografi.

Pengolahan data presensi adalah suatu proses kegiatan pencatatan terhadap setiap presensi dengan tujuan untuk mengetahui data dan laporan berkaitan dengan presensi secara periodik. Di dalam sistem data presensi dilakukan beberapa operasi data antara lain: pencatatan, pemeriksaan, penggolongan, penyusunan, dan penyimpanan. Untuk memperlancar pelaksanaan tersebut diperlukan program yang cepat, efektif, dan mudah dalam penggunaannya. (Kristanto, 2008).

Pembuatan aplikasi mengenai aplikasi presensi juga sudah pernah dilakukan sebelumnya oleh Rahayu, et. al. (2015) dalam jurnal yang berjudul “Perancangan Aplikasi Absensi Peserta Bimbingan Belajar berbasis Web dengan menggunakan

framework YII". Penelitian tersebut bertujuan untuk mengembangkan Sistem Informasi yang bertujuan untuk mengetahui apakah siswa sakit, ijin, alpa, atau terlambat untuk setiap kelas.

Penelitian serupa juga dilakukan oleh Indah dan Dedy (2015), dalam jurnalnya presensi sidik jari (*fingerprint*) berbasis *Web Service* fakultas MIPA Universitas Mulawarman. Penelitian ini bertujuan untuk membangun sebuah sistem yang mengelola presensi pegawai menggunakan bahasa pemrograman PHP (PHP: *Hypertext Prepocessor*) yang dapat menjadi alat bantu rekapitulasi kehadiran pegawai. Penelitian ini menghasilkan sebuah aplikasi presensi yang merepresentasikan data kehadiran secara cepat dan mudah, sehingga data ini dapat digunakan untuk keperluan selanjutnya seperti perhitungan uang makan bulanan. Sehingga dengan sistem tersebut diharapkan dapat menghindari terjadinya korupsi waktu yang sering dilakukan dengan cara "menitip" presensi kepada orang lain dan dengan alat ini diharapkan akan meningkatkan kedisiplinan di lingkungan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Mulawarman.

Penelitian lain juga dilakukan oleh Norhikmah, et. al. (2016) STMIK AMIKOM Yogyakarta yang berjudul penggunakan *QR (Quick Response) code* dalam presensi berbasis *android*, mungungkapkan bahwa *smartphone* berbasis *android* dapat membantu penggunanya dalam melakukan aktivitas yang sebelumnya sulit untuk dilakukan. Salah satunya adalah untuk melakukan presensi, mayoritas masih menggunakan cara manual, yaitu dengan mengedarkan kertas presensi yang ditandatangani oleh mahasiswa. Sehingga *smartphone* berbasis *android* juga mampu digunakan untuk melakukan presensi oleh dosen yang

bersangkutan dan langsung di-*export* dalam bentuk *file micorosft excel.* Dari hasil penelitian tersebut diketahui bahwa dalam proses presensi kita dapat menggunakan *smartphone* berbasis *android* sebagai media untuk presensi.

Penelitian lain juga dilakukan oleh Ardiyanto dan Santiko STMIK Amikom Purwokerto dengan judul rancangan bangun sistem pengolahan presensi siswa di SMK Wiworotomo Purwokerto Tujuan dari penelitian ini adalah menghasilkan sebuah sistem informasi yang dapat melakukan presensi menggunakan *Barcode* di SMK Wiworotomo Purwokerto. Pengembangan sistem menggunakan RAD (*Rapid Application Development*). Berdasarkan penelitian tersebut, dapat disimpulkan bahwa proses pembuatan sistem presensi siswa dengan menggunakan *Barcode* dapat diterapkan di SMK Wiworotomo Purwokerto dikarenakan sistem presensi siswanya sedang dalam proses pengembangan dari manual ke komputerisasi.

Berdasarkan pengujian dan analisis yang telah dilakukan oleh masing-masing peneliti, keempat peneliti tersebut membangun beberapa presensi menggunakan beberapa media. Namun masih terdapat kelemahan pada keempat penelitian tersebut. Penelitian yang dilakukan oleh Sri Rahayu, et. al. (2015) memiliki kelemahan terhadap *security* yang digunakan. *Web* yang digunakan hanya mempunyai satu autentikasi yaitu login sebagai peserta, sehingga peserta lain akan dapat dengan mudah untuk menitip presensi kepada temannya yang lain. Kemudian penelitian yang dilakukan oleh Indah dan Dedy (2015), mempunyai kelemahan pada pengadaan alat *fingerprint* yang terlalu mahal. Sehingga, sangat tidak efektif jika diterapkan pada sebuah instansi besar seperti universitas. Penelitian yang dilakukan oleh Norhikmah, et. al. (2016), mempunyai kelemahan pada proses

presensi yang dilakukan dimana data presensi harus diubah lagi kedalam bentuk *file microsoft excel.* Hal ini kurang efektif karena staf harus memasukkan rekap presensi ulang kedalam sistem. Kemudian penelitian yang dilakukan oleh Ardiyanto dan Santiko memiliki kelemahan dalam hal proses presensi yang terlalu lama jika harus memindai *barcode* satu per satu. Hal ini tentu akan sangat tidak efektif jika diterapkan di dalam kelas yang mempunyai mahasiswa lebih dari 40 orang.

Melihat potensi token sebagai pengaman, maka token juga dapat dikembangkan pada sistem presensi menggunakan *smartphone* berbasis *android*. Dengan menggunakan teknologi *web service,* sistem presensi online dapat dikembangkan pada dua *client,* yaitu dosen dan mahasiswa. *Client* untuk dosen digunakan untuk membuat *token* sedangkan *client* untuk mahasiswa digunakan untuk memasukkan *token*. Sehingga mahasiswa dapat melakukan presensi menggunakan *token* yang telah dibuat oleh dosen.

2.2 Landasan Teori

2.2.1 Pengelolaan Data Presensi

Pengolahan data presensi adalah suatu proses kegiatan pencatatan terhadap setiap presensi dengan tujuan untuk mengetahui dan laporan berkaitan dengan data presensi. Didalam sistem data presensi dilakukan beberapa operasi data antara lain: pencatatan, pemeriksaan, penggolongan, penyusunan, dan penyimpanan (Kristanto, 2008). Tujuan dari pengolahan data presensi adalah untuk mengetahui data dan informasi yang berkaitan dengan informasi presensi.

2.2.2 *One-Time Password*

Menurut Kalaikavitha (2013), *One Time Password* (OTP) merupakan teknik autentikasi yang hanya berlaku untuk satu kali penggunaan. OTP menjamin keamanan terhadap serangan *replay*, dimana penyerang yang berhasil mendapatkan nilai OTP yang sudah digunakan tidak mungkin dapat digunakan kembali. OTP menggunakan algoritma pembangkitan dengan sifat semu acak yang telah diuji secara kiptografi. Metode-metode yang digunakan pada pembangkitan OTP yaitu:

a. Berdasarkan sinkronisasi waktu, metode seperti ini digunakan untuk autentikasi antara *client* dan *server* karena hanya berlaku untuk waktu yang singkat.
b. Berdasarkan suatu algoritma matematika yang menggunakan masukkan nilai OTP sebelumnya untuk mendapatkan nilai OTP baru.

Berdasarkan suatu nilai *challenge* dimana OTP dihasilkan dari suatu algoritma matematika yang mengkombinasikan pengetahuan terhadap nilai rahasia *challenge* (contoh metode seperti ini diterapkan pada autentikasi untuk suatu transaksi atau counter).

2.2.3 *Security Token*

Menurut Adikusuma, et. al. (2009), *Security token* merupakan objek fisik untuk autentikasi sebuah sistem. Jenis *security token* yang sering digunakan adalah *security token* untuk infrastruktur kunci publik, *One time password* (OTP), dan *communication means*.

Security token yang termasuk ke dalam infrastruktur kunci publik berisi data identitas pengguna yang digunakan untuk tanda tangan digital. Sedangkan security *token* yang termasuk jenis OTP digunakan untuk menghasilkan sandi lewat dan hanya dapat digunakan satu kali karena sandi ini akan terus berubah, dan yang termasuk jenis *communication means* mempunyai kemampuan untuk mentransmisikan datanya kepada *server* (Adikusuma, et. al. 2009). Sampai saat ini sebagian besar *token* yang banyak digunakan untuk fasilitas internet adalah *token* yang termasuk dalam jenis OTP yang mebangkitkan barisan acak berdasarkan waktu. Dalam hal ini, dibutuhkan sinkronisasi waktu antara waktu pada *token* dan waktu pada *server* (Barus, 2007).

Karakter didalam *token* dikategorikan sebagai salah satu dari lima kelas *token* yang menggambarkan fungsi mereka (*identifier, keyword, literal string,* operator, separator) sesuai dengan aturan bahasa pemrograman.

Menurut Oktafia (2008), *token* merepresentasikan nama:

1. *Identifier* (nama variabel, fungsi dan tipe atau nama yang didefinisikan pemakai).
2. *Keyword*
3. *Literal String*
4. Operator
5. Label
6. Simbol tanda (tanda kurung, koma dan titik koma).

Dalam penelitian ini jenis *security token* yang digunakan adalah *security token* untuk *One time password* (OTP) dan hanya menggunakan beberapa karakter dalam penerapannya, yaitu:

1. Angka (1, 2, 3, 4, 5, 6, 7, 8, 9, 0)
2. Huruf besar (A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z)
3. Huruf kecil (a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z)

Token ini terdiri dari delapan digit random *keyword*. Proses *generate random token* terjadi setelah pengguna (dosen) membuat sesi baru pada saat perkuliahan akan dimulai. Setelah sesi dibuat, sistem akan melakukan *random* dari daftar karakter yang disebutkan. Menurut Wicaksono, A. (2016), sistem akan melakukan pengecekkan kedalam *database* apakah *token* sudah pernah dibuat untuk sesi lain pada hari yang sama atau belum. Jika *token* yang dibuat sudah digunakan oleh sesi yang lain dan dihari yang sama, maka sistem akan melakukan pengulangan dalam proses *generate random token* sampai didapatkan *token* yang belum digunakan pada sesi yang lain untuk hari yang sama. Setelah sistem mendapatkan *token* yang unik (belum digunakan dihari yang sama pada saat sesi perkuliahan dibuat), *token* akan ditampilkan pada halaman detail sesi perkuliahan. Sehingga *token* ini nantinya akan digunakan oleh mahasiswa untuk melakukan presensi.

2.2.4 *Two Factor Authentication*

Autentikasi adalah suatu proses untuk pembuktian suatu identitas (Kessler, 1998). Tujuan dari autentikasi adalah untuk membuktikan keabsahan dari seseorang

dalam menggunakanan suatu layanan. Menurut Pamadya (2011), metode autentikasi bisa dilihat dalam 4 kategori metode:

1. *Something you know*, merupakan metode yang paling umum dengan mengandalkan *password*.
2. *Something you have*, merupakan faktor tambahan untuk membuat autentikasi menjadi lebih aman dengan mengandalkan barang yang sifatnya unik, contohnya: kartu magnetik (*smart card*), USB *token*, *token hardware*, dan sebagainya.
3. *Something you are*, merupakan metode yang mengandalkan keunikan bagian-bagian tubuh yang tidak mungkin ada pada orang lain, seperti sidik jari, sidik retina dan suara.
4. *Something you do*, melibatkan bahwa setiap *user* dalam melakukan sesuatu dengan cara yang berbeda. Contoh: Penggunaan analisis suara (*voice recognation*), dan analisis tulisan tangan.

Pada aplikasi yang sensitif seperti transaksi keuangan, satu metode autentikasi saja tidak cukup. Oleh karena itu muncul istilah 2FA (*Two Factor Authentication*), 2FA merupakan sistem autentikasi yang menggunakan 2 faktor (metode) yang berbeda. Metode autentikasi dapat dikombinasikan untuk meningkatkan keamanan, salah satu contohnya adalah dengan kombinasi "*something you have*" berupa kartu ATM dengan "*something you know*" berupa PIN. Kombinasi ini merupakan kombinasi yang paling banyak dipakai.

2.2.5 *UML*

Unified Modelling Language (UML) adalah alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual (Braun, et. al. 2001). Selain itu merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek (Whitten, et. al. 2004).

Penggunaan UML bertujuan untuk mengidentifikasikan bagian-bagian yang termasuk dalam lingkup sistem didalam aplikasi, mendokumentasikan hasil analisa dan desain serta untuk menggambarkan sebuah sistem *software*. Model UML yang dipakai dalam pengembangan aplikasi ini antara lain *Use Case* Diagram dan *Activity* Diagram.

a. *Use Case* Diagram

Use Case Diagram mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem yang akan dibuat. Dapat dikatakan *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

b. *Activity* Diagram

Activity Diagram merupakan diagram yang digunakan untuk menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.

2.2.6 *Mobile Application*

Menurut Buyens (2001) *mobile application* berasal dari dua kata yaitu *application* dan *mobile*. Secara istilah *application* adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna, sedangkan *mobile* dapat diartikan sebagai perpindahan dari suatu tempat ke tempat yang lain.

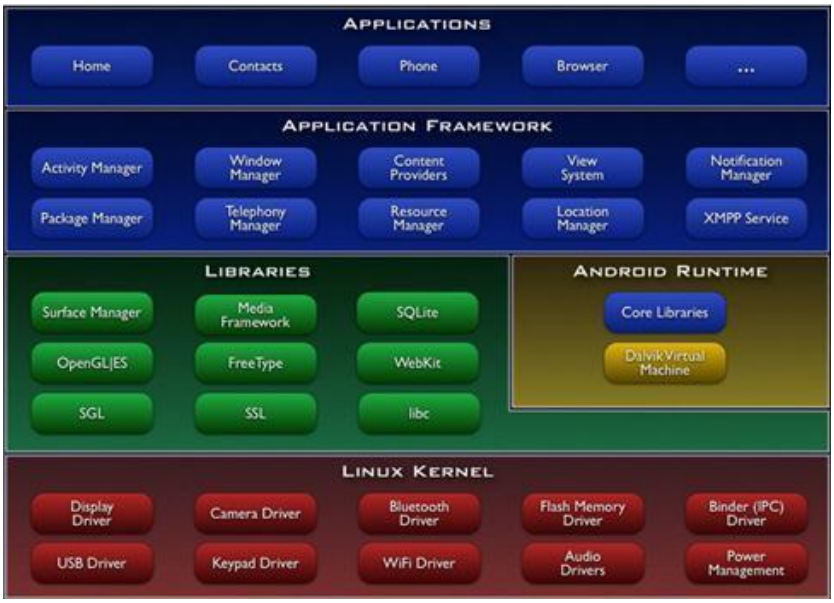
Kata *mobile* mempunyai arti bergerak atau berpindah, sehingga *application mobile* menurut Rangsang Purnama (2010), adalah istilah untuk aplikasi yang berjalan di *mobile device*. Dengan menggunakan *mobile application*, pengguna dapat dengan mudah melakukan berbagai macam aktifitas mulai dari, belajar, mengerjakan pekerjaan kantor, *browsing* dan lain sebagainya.

2.2.7 *Android*

Android adalah sistem operasi berbasis Linux yang dirancang untuk *smart phone* dan komputer tablet. *Android* awalnya dikembangkan oleh *Android*, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya *Open Handset Alliance*, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Arsitektur *android* dapat dilihat pada gambar 2.1.

Gambar 2.1 merupakan susunan dari arsitektur pada *android* yang terdiri dari *Linux Kernel*, *Library*, *Application Framework*, dan *Applications*. *Linux kernel*

menyediakan *display driver, USB driver, camera driver, keypad River, Bluetooth driver, WiFi driver, Flash Memory driver, audio driver, Bonder (IPC) driver* dan *power management* sebagai pengatur sistem operasi. (Damarullah, et. al. 2013)



Gambar 2.1 Arsitektur Platform *Android*

Library terdiri dari *Surface Manager, Media Framework, SQLite, OpenGL| ES, Free Type, WebKit, SGL, SSL* dan *libc*, kemudian *Android Runtime* yang terdiri dari dua bagian utama, yakni *core libraries* dan *dalvik virtual machine*. Fungsi dari *library* sendiri adalah untuk mengarahkan perangkat *android* dalam menangani berbagai tipe data.

Application Framework terdiri dari *activity manager, windows manager, content provider* dan *view system*. Lapisan ini berfungsi untuk memanajemen fungsi dasar dari *android*, seperti memanajemen *resource*, panggilan, tampilan, dan pemberitahuan.

Applications yang merupakan lapisan terluar dari arsitektur andorid. Pengguna hanya akan melihat program ketika digunakan tanpa mengetahui proses yang terjadi dibalik lapisan aplikasi. Lapisan ini berjalan dalam *Android runtime* dengan menggunakan *class* dan *service* yang tersedia pada *framework* aplikasi. (Damarullah, et. al. 2013)

2.2.8 Eclipse ADT *(Android Development Tools)*

ADT adalah sebuah *plugin* untuk Eclipse yang menyediakan *tools* yang terintergrasi dengan Eclipse. ADT menyediakan fitur yang membantu untuk mengembangkan aplikasi *Android* seperti, untuk pembuatan tampilan aplikais, dan pembuatan program yang berjalan pada sistem operasi *android*. (Damarullah, et. al. 2013)

2.2.9 *Android* SDK

Android SDK merupakan tools untuk programmer yang ingin mengembangkan aplikasi berbasis *android*. *Android* SDK terdiri dari *debugger*, *libraries, handset emulator*, dokumentasi, contoh kode, dan tutorial. (Damarullah, et. al. 2013)

2.2.10 *Software Development Life Cycle (SDLC)*

SDLC adalah kerangka yang mendiskripsikan sejumlah kegiatan yang dilakukan menggunakan beberapa tahap pada saat pembuatan sebuah *software* (Fatwa, 2007). Selain itu, SDLC juga penting pada saat melakukan kegiatan *maintenance software*.

Model SDLC yang dipakai dalam penelitian ini adalah model *Waterfall*. Menurut (Sommerville, 2011), *waterfall* model adalah sebuah contoh dari proses perencanaan dimana semua proses kegiatan harus terlebih dahulu direncanakan dan dijadwalkan sebelum dikerjakan. *Waterfall* Model atau *Classic Life Cycle* merupakan model yang paling banyak dipakai dalam *Software Engineering* (SE). Menurut Bassil (2012), disebut *waterfall* karena tahap demi tahap yang harus dilalui menunggu selesainya tahap sebelumnya dan berjalan berurutan.

2.2.11 *Black Box Testing*

Terdapat beberapa metode dalam melakukan pengujian perangkat lunak, yaitu dengan menggunakan metode *white box testing*, *black box testing*, dan g*rey box testing.* Dari beberapa metode tersebut, pada skripsi ini dipilihlah *black box testing* dalam melakukan pengujian perangkat lunak. *Black box testing* dipilih karena dianggap lebih tepat dibandingkan dengan *white box testing* maupun *grey box testing*. Dalam pembuatan perangkat lunak, diperlukan testing atau pengujian untuk mencari kesalahan fungsi – fungsi dalam perangkat lunak, sehingga dalam hal ini *black box testing* lebih sesuai. Pengujian ini digunakan untuk mengetahui apakah fungsi-fungsi dalam perangkat lunak sudah sesuai dengan yang diharapkan.

Menurut Roger S. Pressman (2010), *black box testing* berfokus pada persyaratan fungsional perangkat lunak yang memungkinkan engineer untuk memperoleh input yang sepenuhnya akan melaksanakan persyaratan fungsional untuk sebuah program. *Black box testing* berusaha untuk menemukan kesalahan dalam kategori berikut:

a. Fungsi yang tidak benar atau fungsi yang hilang.

b. Kesalahan antarmuka.

c. Kesalahan dalam struktur data atau akses database eksternal.

d. Kesalahan perilaku (*behavior*) atau kesalahan kinerja.

e. Inisialisasi dan pemutusan kesalahan.