

## BAB II TINJAUAN PUSTAKA

### **2.1 Push Notifikasi**

*Push* Notifikasi adalah sebuah layanan yang banyak digunakan untuk keperluan pemberitahuan melalui pesan pendek yang ada di *smartphone*. Dengan adanya layanan *Push* Notifikasi tersebut, pengguna dapat terbantu dalam hal yang bersifat pemberitahuan secara singkat. Pada implementasinya *Push* Notifikasi dapat dimanfaatkan dalam berbagai keperluan sehari-hari misalnya untuk *monitoring* absensi, *update* berita terbaru, dan sebagainya. Aplikasi yang akan dirancang adalah sebuah aplikasi yang dapat mengirim *Push* Notifikasi yang nantinya akan dapat dikembangkan di berbagai bidang sesuai dengan kebutuhan pengguna. Kurangnya pengetahuan dalam memberikan informasi secara *real time*, berakibat informasi yang tersampaikan tidak *up-to-date*, sehingga dalam berbagai situasi dan kondisi informasi yang diberikan telah usang. *Push* Notifikasi salah satu layanan yang dapat menjawab masalah tersebut sehingga tidak ada lagi informasi yang terbaru tidak tersampaikan, dengan penggunaan layanan ini setiap terjadi *update* informasi maka akan langsung terkirim sebagai pesan notifikasi, sehingga informasi yang terbaru tidak akan terlewatkan. Layanan *Push* Notifikasi umumnya banyak diterapkan pada aplikasi mobile seperti Android dan IOS. Untuk penggunaan Sistem Operasi *mobile* terbesar berdasarkan *statcounter* untuk tahun 2012 sampai 2016 di Indonesia dikuasi oleh Sistem Operasi Android (Sylvia & Kurniawan, 2019).

### **2.2 Pengertian Round Robbin**

*Round Robbin* adalah penjadwalan proses menerapkan strategi *pre-emptive*, bukan di *pre-emptive* oleh proses lain, tapi terutama oleh penjadwal berdasarkan jatah waktu pemroses yang disebut kwanta (quantum). *Round Robin* disebut juga *Fair Time Scheduling*, memiliki prinsip dasar, yaitu semua sumber antrian dianggap sama sehingga diberi waktu yang disebut *time quantum*. Jika *time quantum* habis atau proses selesai, maka proses berlanjut ke antrian berikutnya. Penjadwalan ini cukup adil karena tidak ada antrian yang diprioritaskan, semua mendapat jatah

waktu yang sama. Metode *Round Robin* pada dasarnya sama dengan FCFS, hanya saja bersifat preemptive, setiap proses mendapatkan waktu CPU yang disebut dengan waktu quantum (quantum time) untuk membatasi waktu proses, biasanya 1-100 milidetik. Setelah waktu habis, proses ditunda dan ditambahkan pada *ready queue* (Sylvia & Kurniawan, 2019).

### 2.3 Android

Android merupakan sistem operasi *smartphone* yang sangat populer karena bersifat *open source* yang menjadi magnet bagi para *developer* untuk mengembangkan aplikasi-aplikasinya. Android awalnya dikembangkan oleh Android, inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem informasi 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. Ponsel android pertama mulai dijual pada bulan oktober 2008 (Arfida & Wibowo, 2018a).

Android merupakan salah satu OS ( Sistem Operasi ) yang banyak diterapkan di berbagai *smartphone*. Android ini juga mempunyai berbagai versi dari awal mula terbentuknya Android sampai sekarang, berikut daftar nama versi android dari awal :



Gambar 2. 1 Nama dan Versi Android

Tabel 2.1. Nama-nama versi android

Versi Android	Nama Android
Android versi 1.0	<b>Apple Pie / Alpha</b>
Android versi 1.1	<b>Beta</b>
Android versi 1.5	<b>CupCake</b>
Android versi 1.6	<b>Donut</b>
Android versi 2.0	<b>Éclair</b>
Android versi 2.2	<b>Froyo</b>
Android versi 2.3	<b>Gingerbread</b>
Android versi 3.0	<b>Honeycomb</b>
Android versi 4.0	<b>Ice Cream Sandwich</b>
Android versi 4.1	<b>Jelly bean</b>
Android versi 4.4	<b>KitKat</b>
Android versi 5.0	<b>Lollipop</b>
Android versi 6.0	<b>Marsmelow</b>
Android versi 7.0	<b>Nougat</b>
Android versi 8.0	<b>Oreo</b>
Android versi 9.0	<b>Pie</b>

## 2.4 Perangkat Lunak Pengembangan Sistem

Untuk membangun aplikasi Sistem informasi berbasis android diperlukan beberapa perangkat lunak yang digunakan dalam membangun aplikasi tersebut. Beberapa perangkat lunak yang digunakan adalah sebagai berikut :

### 2.4.1 Android Studio

Android Studio merupakan sebuah software tools *Integrated Development Environment* (IDE) untuk platform Android. Android studio ini diluncurkan pada tanggal 16 Mei 2013 pada konferensi Google I/O oleh produk manajer Google, Ellie Powers (Wijayanto, 2017).

### 2.4.2 Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi aplikasi berbasis java umumnya dikompilasi ke dalam *p-code (bytecode)* dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus di *desain* untuk memanfaatkan dependensi implementasi seminimal mungkin (Irsan, 2015).

### 2.4.3 Xampp

XAMPP merupakan *tool* yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan menginstal XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server apache*, PHP, dan MySQL secara manual. XAMPP akan menginstalasi dan mengkonfigurasi secara otomatis (Achmad Nuzul Mariyus, Neni Purwati, 2019)

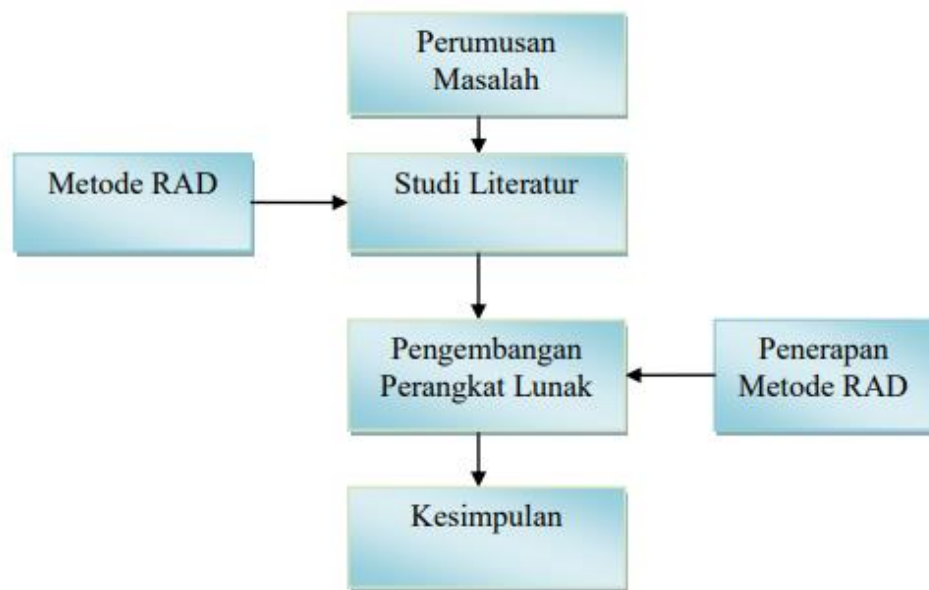
### 2.4.4 MySQL

MySQL adalah merupakan software yang tergolong database server dan bersifat *opensource*. *Opensource* menyatakan bahwa software ini dilengkapi dengan *sourcecode* (kode yang dipakai untuk membuat MySQL), selain tentu saja bentuk *executable*-nya atau kode yang dapat dijalankan secara langsung dalam sistem operasi, dan bisa diperoleh dengan cara mengunduh di internet secara gratis. Database adalah sekumpulan (sering saling terkait) data, baik teks, angka, atau file biner yang disimpan dan diselenggarakan oleh DBMS. MySQL sebagai server database *open source* yang digunakan pada aplikasi terutama dalam membuat web, MySQL digunakan dalam mengolah data yang terdapat pada database.(Achmad Nuzul Mariyus, Neni Purwati, 2019)

### 2.4.5 Visual Studio Code

Microsoft Visual Studio adalah sebuah *Integrated Development Environment* buatan Microsoft Corporation. Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di atas Windows) ataupun *managed code* (dalam bentuk Microsoft Intermediate Language di atas .NET Framework). (Putri & Azpar, 2016)

## 2.5 Metode pengembangan perangkat lunak



Gambar 2. 2 Metode *Rapid Application Development*

### 2.5.1 Metode *Rapid Application Development* (RAD)

*Rapid Application Development* (RAD) merupakan metode pengembangan sistem yang memiliki keunggulan karena tahapan yang singkat dan cepat seperti tahapan *Requirements Planning* untuk mengidentifikasi tujuan dari aplikasi atau sistem, tahapan RAD *Design Workshop* (pemodelan) untuk membangun tampilan visual desain dan alur kerja pengguna dan tahapan implementasi untuk pembangunan sistem dan pengujian. Sehingga dengan tahapan-tahapan tersebut, penerapan

metode RAD sangat tepat dan sesuai dalam pembangunan sistem berbasis website (Hiswara, Fitriyani, & Nugraha, 2020).

Pemilihan metode pengembangan RAD dikarenakan RAD mengikuti tahapan pengembangan sistem seperti umumnya, tetapi mempunyai kemampuan untuk menggunakan kembali komponen yang ada (*Reusable Object*). Selain itu, juga Setiap fungsi dapat dimodulkan dalam waktu tertentu dan dapat dibicarakan oleh tim RAD yang terpisah dan kemudian diintegrasikan sehingga waktunya lebih efisien.

Adapun tahapan- tahapan berdasarkan metode *Rapid Application Development* (RAD) adalah sebagai berikut :

#### 1. *Requirements Planning* (Perencanaan Persyaratan)

Tahapan *Requirements Planning* bertujuan untuk mengidentifikasi kebutuhan, batasan dan objektifitas dari sistem yang akan dibangun, dengan cara mengumpulkan data dari *stakeholder*. Aktivitas yang dilakukan dengan melakukan pengamatan langsung dan mengumpulkan data dari buku-buku dan jurnal-jurnal yang menunjang dan relevan. Hasil yang didapatkan berupa mekanisme atau prosedur pengambilan data penelitian dan spesifikasi kebutuhan sistem.

#### 2. *RAD Design Workshop* (Pemodelan)

Tahapan *RAD Design Workshop* bertujuan untuk merancang semua kegiatan dalam arsitektur sistem secara keseluruhan dengan melibatkan identifikasi dan deskripsi abstraksi sistem perangkat lunak yang mendasar dan hubungan-hubungannya. Aktivitas yang dilakukan dengan melakukan identifikasi pelaku, analisis proses dan kinerja sistem, mengidentifikasi struktur objek dan relasinya, pemodelan interaksi obyek dan *behavior*, dan mendesain antarmuka. Hasil yang didapatkan berupa pemodelan sistem.

### 3. Implementasi

Tahap implementasi bertujuan untuk mengimplementasikan metode program sesuai dengan kebutuhan sistem. Aktivitas yang dilakukan dengan membangun sistem sesuai dengan pemodelan yang dibangun. Hasil yang didapatkan berupa sistem pengajuan pengambilan data penelitian berbasis website dengan tahapan – tahapan berdasarkan metode *Rapid Application Development (RAD)*.

#### **2.6 Black Box Testing**

Menurut Mustaqbal et al., (2015) *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. *Tester* dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. *Black Box Testing* bukanlah solusi alternatif dari *White Box Testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing*. *Black Box Testing* cenderung untuk menemukan hal-hal berikut:

1. Uji *Interface*.
2. Uji fungsi menu dan tombol.

#### **2.7 Unified Modelling Language (UML)**

*Unified Modelling Language (UML)* adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain (Munawar, 2018).

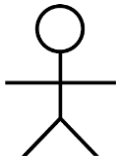
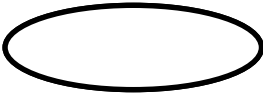




Adapun beberapa jenis UML yang dapat membantu perancangan sistem adalah sebagai berikut:

##### **2.7.1 Use Case Diagram**

*Use Case diagram* merupakan deskripsi peringkat tinggi bagaimana perangkat lunak(aplikasi) akan digunakan oleh penggunanya(Arfida & Wibowo, 2018b). Secara lebih spesifik *use case diagram* digunakan untuk mengumpulkan

kebutuhan dari sebuah sistem baik karena pengaruh internal maupun eksternal. *Use Case* adalah abstraksi dari interaksi antara sistem dan *actor*. Oleh karena itu sangat penting memilih abstraksi yang cocok. *Use Case* juga adalah alat bantu terbaik guna menstimulasi pengguna potensial untuk mengatakan tentang suatu sistem dari sudut pandangnya.

Tabel 2. 2 Simbol *Use case diagram*



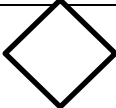

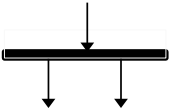
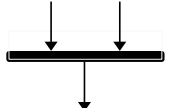
Simbol	Keterangan
 <i>Actor</i>	<b>Actor</b> : Mempresentasikan seseorang atau sesuatu(seperti perangkat,sistem lain) yang berinteraksi dengan sistem. <i>Actor</i> hanya berinteraksi dengan use case tetapi tidak memiliki kontrol atas <i>use case</i> .
	<b>Use Case</b> : Adalah gambaran fungsionalitas dari suatu sistem, sehingga <i>customer</i> atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.
	<b>Association</b> : Menghubungkan link antar <i>element</i> .
	<b>Generalization</b> : Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk.
	<b>Include</b> : Yaitu kelakuan yang harus terpenuhi agar sebuah <i>event</i> dapat terjadi, dimana pada kondisi ini sebuah <i>use case</i> adalah bagian dari <i>use case</i> lainnya.
	<b>Extend</b> : Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang di berikan.



### 2.7.2 Activity Diagram

*Activity diagram* adalah bagian penting dari UML yang menggambarkan aspek dinamis dari sistem. Logika procedural, proses bisnis dan aliran kerja suatu bisnis bisa dengan mudah dideskripsikan dalam *activity diagram*. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flowchart* tidak bisa.




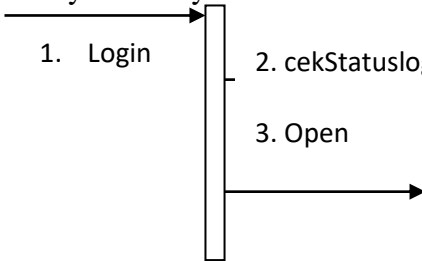
Tabel 2. 3 Simbol *Activity Diagram*



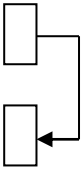

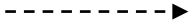

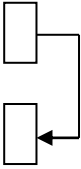
Simbol	Keterangan
 <b>Status Awal</b>	Status awal aktivitas <i>system</i> , sebuah diagram aktivitas memiliki sebuah status awal.
 <b>Aktivitas</b>	Aktivitas yang dilakukan <i>system</i> , aktivitas biasanya diawali dengan kata kerja.
 <b>Percabangan</b>	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
 <b>Status Akhir</b>	Status akhir yang dilakukan <i>system</i> , sebuah diagram aktivitas, sebuah diagram aktivitas memiliki sebuah status akhir.
 <b>Percabangan</b>	Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel
 <b>Penggabungan</b>	Digunakan untuk kegiatan yang digabungkan.


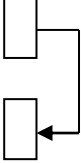

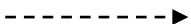

### 2.7.3 Sequence Diagram

*Sequence diagram* digunakan untuk menggambarkan perilaku pada sebuah scenario. Diagram ini menunjukkan sejumlah contoh obyek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam *use case*.

Tabel 2. 1 Simbol *Sequence Diagram*

Simbol	Deskripsi
<p>Aktor/ <i>actor</i></p>  <p>nama aktor</p> <p>Atau</p> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">Nama Aktor</div>	<p>Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.</p>
<p>Garis Hidup/<i>lifeline</i></p> 	<p>Menyatakan kehidupan suatu objek</p>
<p>Objek</p> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">Nama actor: Nama Kelas</div>	<p>Menyatakan objek yang berinteraksi pesan</p>
<p>Waktu aktif</p> 	<p>Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif I I adalah sebuah tahapan yang dilakukan didalamnya misalnya</p>  <p>Maka cek Status login () dan open () dilakukan di dalam metode login () actor tidak memiliki waktu aktif.</p>



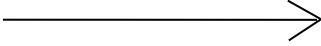
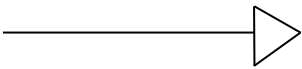
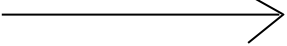
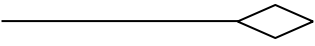
<p>Pesan Tipe Create</p> <p>&lt;&lt;create&gt;&gt;</p> 	<p>Menyatakan suatu objek membuat objek yang lain, arah panah mengarahkan pada objek yang dibuat.</p>
<p>Pesan Tipe Call</p> <p>1: Nama metode ()</p> 	<p>Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri.</p>  <p>1. Nama_metode()</p> <p>Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.</p>
<p>Pesan Tipe Send</p> <p>1: masukan</p> 	<p>Menyatakan bahwa suatu objek mengirimkan data masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.</p>
<p>Pesan Tipe Return</p> <p>1: keluaran</p> 	<p>Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.</p>
<p>Pesan Tipe Call</p> <p>1: Nama metode ()</p> 	<p>Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri.</p>  <p>2. Nama_metode()</p> <p>Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.</p>

<p>Pesan Tipe Call 1: Nama metode ()</p> 	<p>Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri.</p>  <p>3. Nama_metode()</p> <p>Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.</p>
<p>Pesan Tipe Send 1: masukan</p> 	<p>Menyatakan bahwa suatu objek mengirimkan data masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.</p>
<p>Pesan Tipe Return 1: keluaran</p> 	<p>Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.</p>
<p>Pesan Tipe Destory &lt;&lt;destory&gt;&gt;</p> 	<p>Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy.</p>

#### 2.7.4 Class Diagram

*Class diagram* menggambarkan struktur sistem dari kelas-kelas yang akan dibuat untuk membangun sistem (Puspita, Ali, Kasus, & Informatika, (2019). *Class diagram* tidak hanya digunakan untuk memvisualisasikan, menggambarkan, dan mendokumentasikan berbagai aspek sistem tetapi juga untuk membangun kode eksekusi (*executable code*) dari aplikasi perangkat lunak.

Tabel 2. 2 Simbol *Class Diagram*

No.	Simbol	Keterangan
1.	Nama_kelas +Attribute +Operasi	Kelas pada struktur sistem.
2.	Antar Muka/Interface  Nama_Interface	Sama dengan konsep interface dalam pemrograman berorientasi objek.
3.	Asosiasi / <i>Asociation</i> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
4.	Asosiasi Berarah / <i>Directed Association</i> 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
5.	Generalisasi 	Relasi antar kelas dengan makna generalisasipesialisasi (umum khusus)
6.	Ketergantungan / <i>dependency</i> 	Relasi antar kelas dengan makna ketergantungan antar kelas.
7.	Agregasi / <i>aggregation</i> 	Relasi antar kelas dengan makna semua bagian ( <i>whole-part</i> )

## 2.8 Penelitian Terkait

Adapun penelitian terdahulu dapat dilihat pada tabel dibawah ini :

NO	JUDUL	PENELITI	KETERANGAN
1.	Aplikasi Mobile Untuk Notifikasi Kegiatan Dosen Menggunakan Metode <i>Round-Robbin</i> (Study Kasus : IIB DARMAJAYA)	Ayu Sylvia (2018)	Aplikasi untuk membantu pendistribusian perubahan informasi akademik secara Realtime dan mengingatkan dosen untuk melaksanakan kegiatan akademik sesuai jadwal. Penelitian yang dilakukan oleh penulis bertujuan untuk membangun aplikasi yang dapat digunakan sebagai pengingat jadwal kegiatan akademik di IIB DARMAJAYA. Penelitian ini terbagi dalam dua tahap besar, pertama analisis dan Perancangan aplikasi, kedua pembangunan aplikasi. Penelitian ini akan mengembangkan sistem <i>Platform</i> . Platform Mobile yang digunakan Dosen untuk melihat beberapa informasi agenda akademik dan menerima informasi atau pengumuman <i>realtime</i> dan notifikasi atau alarm terkait dengan kegiatan akademik kepada pengguna aplikasi

			<i>mobile.</i>
2.	IMPLEMENTASI SISTEM INFORMASI PRESENSI DOSEN BERBASIS <i>CLIENT SERVER</i> (Studi Kasus: Institut Informatika dan Bisnis Darmajaya)	Nurfiana (2016)	Sistem informasi presensi dosen mengajar berbasis <i>client server</i> . Studi kasus dari penelitian ini adalah sistem presensi dosen mengajar pada Institut Informatika dan Bisnis (IIB) Darmajaya. Perekaman data pada IIB Darmajaya dilakukan dengan menggunakan dua perekaman data, yaitu aplikasi presensi yang berada pada petugas PLPP dan buku presensi mengajar. meskipun perekaman data presensi telah menggunakan 2 (dua) cara perekaman, namun kemungkinan kecurangan antara dosen dan petugas PLPP dengan menitipkan tanda tangan presensinya baik melalui aplikasi maupun buku presensi mengajar. Pengembangan sistem presensi yang terintegrasi atau <i>client server based</i> diperlukan untuk mengatasi kecurangan dari para dosen dan sekaligus sebagai sistem presensi terbaru. Dalam penelitian ini juga akan

			dipaparkan teori-teori yang digunakan, serta perancangan sistem informasi. Sehingga diharapkan hasil yang didapat memberikan solusi dalam perubahan sistem presensi yang lama.
3.	PERANCANGAN APLIKASI <i>PUSH NOTIFICATION</i>	(Siddik & Nasution, 2018)	Aplikasi yang akan dirancang adalah sebuah aplikasi yang dapat mengirim <i>Push Notifikasi</i> yang nantinya akan dapat dikembangkan di berbagai bidang sesuai dengan kebutuhan pengguna. Kurangnya pengetahuan dalam memberikan informasi secara <i>realtime</i> , berakibat informasi yang tersampaikan tidak <i>up-to-date</i> , sehingga dalam berbagai situasi dan kondisi informasi yang diberikan telah usang