

SKRIPSI

KOLEKTOR PENGUMUMAN INFORMATIKA



Ellena Angelica

NPM: 2015730029

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2019**

UNDERGRADUATE THESIS

INFORMATICS ANNOUNCEMENT COLLECTOR



Ellena Angelica

NPM: 2015730029

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2019**

LEMBAR PENGESAHAN

KOLEKTOR PENGUMUMAN INFORMATIKA

Ellena Angelica

NPM: 2015730029

Bandung, 22 Mei 2019

Menyetujui,

Pembimbing

Elisati Hulu, M.T.

Ketua Tim Penguji

Anggota Tim Penguji

Natalia, M.Si.

Dr. Veronica Sri Moertini

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

KOLEKTOR PENGUMUMAN INFORMATIKA

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 22 Mei 2019

Meterai Rp. 6000

Ellena Angelica
NPM: 2015730029

ABSTRAK

Pengumuman di jurusan Teknik Informatika UNPAR pada umumnya menggunakan *email*. *Email* memiliki kekurangan, yaitu kurang terorganisir. Skripsi ini mencoba untuk mengatasi kekurangan itu dengan mengumpulkan pengumuman dari *email* di satu tempat. Tempat yang dipakai di skripsi ini adalah BlueTape. BlueTape adalah *website* milik FTIS yang berfungsi untuk menangani urusan-urusan di FTIS UNPAR. Solusi dari kekurangan *email* dibuat dengan cara membuat fitur kolektor pengumuman di BlueTape.

Fitur kolektor pengumuman akan melakukan sinkronisasi *email* untuk mengumpulkan *email* yang berisi pengumuman di ruang lingkup jurusan Informatika menggunakan PHP IMAP. Setelah memastikan pengumuman dilakukan oleh pihak yang berwenang, pengumuman akan ditampilkan di BlueTape. Dengan memanfaatkan layanan LINE@, mahasiswa jurusan Informatika akan diberitahu apabila ada pengumuman baru di BlueTape.

Pengujian pada skripsi ini dilakukan dua kali. Pengujian pertama dilakukan dengan mengirimkan berbagai macam *email* ke alamat email milik BlueTape. Pengujian kedua dilakukan dengan melibatkan mahasiswa dan dosen secara langsung. Hasil pengujian menunjukkan fitur ini telah berfungsi sesuai harapan, yaitu dapat melakukan sinkronisasi *email*, dapat menampilkan *email* yang berisi pengumuman di BlueTape, dan dapat memunculkan notifikasi pada akun LINE@.

Kata-kata kunci: BlueTape, IMAP, LINE

ABSTRACT

Announcement in UNPAR's Department of Informatics in general using email. Email has a disadvantage, which is less organized. This thesis tries to overcome this shortcoming by collecting announcements from email in one place. The place used in this thesis is BlueTape. BlueTape is FTIS's website which functions to handle matters in UNPAR's FTIS. The solution to the disadvantage of email was made by making an announcement collector feature on BlueTape.

The announcement collector feature will synchronize email to collect email which contains an announcement in the scope of Informatics department using PHP IMAP. After making sure the announcement is made by the authorities, the announcement will be displayed on BlueTape. By utilizing the LINE@ service, Informatics students will be notified when there is a new announcement at BlueTape.

The test in this undergraduate thesis is done twice. The first test was done by sending various kinds of email to the email address of BlueTape. The second test was carried out by involving students and lecturers directly. The test results show that this feature has functioned as expected, that is, it can synchronize email, can display emails containing announcements at BlueTape, and can display notifications on LINE@ accounts.

Keywords: BlueTape, IMAP, LINE

*Untuk diri sendiri, keluarga, teman-teman seperjuangan, dan
semua yang telah mendukung*

KATA PENGANTAR

Puji Tuhan skripsi dengan judul "Kolektor Pengumuman Informatika" ini telah selesai dengan baik dan tepat waktu. Penulis berharap skripsi ini dapat berguna bagi pembaca dan peneliti yang ingin melanjutkan penelitian ini. Terima kasih kepada:

1. Keluarga, khususnya orang tua, yang telah menyemangati saya secara langsung atau tidak langsung saat proses pembuatan skripsi ini.
2. Bapak Pascal Alfadian yang telah membimbing saya di dalam proses pembuatan skripsi ini.
3. Ibu Natalia dan Ibu Veronica Sri Moertini yang telah menguji dan memberi saran untuk skripsi ini.
4. Seluruh mahasiswa kelas ADPL yang telah membantu saya saat proses pengujian eksperimental.
5. Teman-teman seperjuangan saya yang saling mendukung: Evelyn Wijaya, Tegar Muhammad Soekarno, Felicia Christiany, Ferdinandus Renaldi, dan Muhammad Adam Nur Mishwari.
6. Rekan-rekan kerja di perpustakaan UNPAR, khususnya Kak Rysca, Kak Chelsea, Mba Ratna, dan Kak Putri, yang telah menyemangati dan memberi saran kepada saya.
7. Rekan-rekan kerja di DNArtworks yang telah berbagi ilmu yang sangat bermanfaat saat mengerjakan skripsi ini.
8. Teman-teman jurusan Teknik Informatika angkatan 2015.
9. Nama-nama yang belum disebutkan namun membantu saya dalam proses pembuatan skripsi ini.

Bandung, Mei 2019

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	1
1.3 Tujuan	1
1.4 Batasan Masalah	2
1.5 Metodologi	2
1.6 Sistematika Pembahasan	2
2 LANDASAN TEORI	3
2.1 Heroku [1]	3
2.1.1 Arsitektur Heroku	3
2.1.2 Log	11
2.1.3 Deploy Perangkat Lunak	14
2.1.4 Ephemeral Filesystem	17
2.1.5 Basis Data dan Manajemen Data	17
2.1.6 Verifikasi Akun	22
2.2 Cron [2]	22
2.3 Gmail API [3]	23
2.3.1 Resource	23
2.3.2 Scope	24
2.3.3 Penggunaan pada umumnya	24
2.3.4 Implementasi Otorisasi dari Sisi Server	25
2.4 PHP IMAP [4]	26
2.5 LINE [5]	32
2.5.1 LINE Login	32
2.5.2 LINE Bot Designer	33
2.5.3 Clova	33
2.5.4 LINE Pay	34
2.5.5 Messaging API	34
2.5.6 LINE@ Manager	41
2.6 System Usability Scale [6]	41
3 ANALISIS	43
3.1 Analisis Sistem Kini	43
3.1.1 Aturan Kontribusi BlueTape	44

3.1.2	Autentikasi	45
3.1.3	Fitur - Fitur BlueTape	46
3.1.4	Hak Akses	51
3.1.5	Struktur Kelas BlueTape	51
3.2	Analisis Sistem Usulan	52
3.2.1	Diagram <i>Use Case</i>	53
3.2.2	Analisis Heroku	54
3.2.3	Sinkronisasi <i>Email</i>	56
3.2.4	Menghubungkan BlueTape dengan LINE	57
4	PERANCANGAN	59
4.1	Perancangan Kelas	59
4.1.1	Model	60
4.1.2	<i>View</i>	62
4.1.3	<i>Controller</i>	63
4.2	Perancangan Basis Data	66
4.2.1	Tabel Pengumuman	66
4.2.2	Tabel PengumumanLineFollowers	67
4.3	Perancangan Antarmuka	67
4.3.1	Perancangan Antarmuka pada BlueTape	67
4.3.2	Perancangan Antarmuka pada <i>Bot</i> BlueTape	70
5	IMPLEMENTASI DAN PENGUJIAN	71
5.1	Implementasi	71
5.1.1	Lingkungan Pengembangan	71
5.1.2	Implementasi Basis Data	72
5.1.3	Implementasi Kelas	72
5.2	Pengujian	91
5.2.1	Lingkungan Pengujian	91
5.2.2	Pengujian Fungsional	92
5.2.3	Pengujian Eksperimental	95
6	KESIMPULAN DAN SARAN	113
6.1	Kesimpulan	113
6.2	Saran	113
DAFTAR REFERENSI		115
A	KODE PROGRAM	117
B	LAMPIRAN PENGUJIAN EKSPERIMENTAL	129
B.1	Kuesioner	129
B.1.1	Dosen	130
B.1.2	Mahasiswa	131
B.1.3	<i>System Usability Scale</i> dan Saran	133
B.2	Hasil Mentah Kuesioner	135

DAFTAR GAMBAR

2.1	Diagram hubungan antara <i>process type</i> dan <i>dyno</i>	4
2.2	Config vars pada dashboard Heroku	6
2.3	Tabel buildpack heroku	8
2.4	<i>Deploy</i> menggunakan Github <i>Dashboard</i>	16
2.5	<i>Deploy</i> menggunakan Github secara manual	16
2.6	<i>Deploy</i> menggunakan Github secara otomatis	16
2.7	Tabel plan Heroku Postgres	18
2.8	LINE Login	32
2.9	LINE Bot Designer	33
2.10	Situs web Clova	33
2.11	Situs web LINE Pay	34
2.12	Push message dan reply message pada Messaging API	34
2.13	Arsitektur Messaging API	35
2.14	Tampilan LINE Developers <i>console</i> saat login	36
2.15	Tampilan LINE Developers <i>console</i> saat <i>register developer</i>	36
2.16	Tampilan LINE Developers <i>console</i> saat membuat <i>channel</i>	37
2.17	Tampilan LINE Developers <i>console</i> saat konfirmasi pembuatan <i>channel</i>	38
3.1	Tampilan utama BlueTape	43
3.2	Tampilan Cetak Transkrip	46
3.3	Tampilan hasil Request Cetak Transkrip	47
3.4	Tampilan Manajemen Transkrip BlueTape	47
3.5	Tampilan request perubahan kuliah	48
3.6	Tampilan manage perubahan kuliah	48
3.7	Tampilan tambah jadwal dosen	49
3.8	Tampilan jadwal dosen	49
3.9	Tampilan edit jadwal dosen	50
3.10	Tampilan lihat jadwal dosen	50
3.11	Use case diagram fitur kolektor pengumuman	53
4.1	Class diagram fitur kolektor pengumuman	59
4.2	<i>Mockup</i> antarmuka <i>main</i>	68
4.3	<i>Mockup</i> antarmuka <i>read</i>	69
4.4	<i>Mockup</i> antarmuka <i>bot</i> BlueTape	70
5.1	Pengumuman Ujian Eksperimental	95
5.2	Kode QR Bot Shadowtape	95
5.3	<i>Email</i> pada tanggal 24 April 2019 oleh Bapak Pascal	96
5.4	<i>Email</i> pada tanggal 25 April 2019 oleh peneliti	97
5.5	Tampilan pesan LINE setelah pengumuman disebar	98
5.6	Tampilan BlueTape setelah URL dibuka dan pengguna belum <i>login</i>	99
5.7	Tampilan URL pengumuman yang diumumkan oleh Pak Pascal Alfadian	100
5.8	Tampilan URL pengumuman yang diumumkan oleh peneliti	101

5.9	Diagram profil responden	101
5.10	Diagram penggunaan <i>email</i> di kalangan mahasiswa	102
5.11	Diagram penggunaan LINE di kalangan mahasiswa	102
5.12	Diagram mahasiswa yang menerima notifikasi LINE	103
5.13	Diagram mahasiswa yang dapat membuka URL pengumuman	103
5.14	Diagram mahasiswa yang diarahkan kembali ke URL pengumuman setelah <i>login</i>	104
5.15	Diagram dosen yang dapat melihat pengumuman yang diumumkannya	104
5.16	Diagram dosen yang judul pengumumannya sesuai dengan subjek <i>email</i> yang ia kirim	105
5.17	Diagram dosen yang isi pengumumannya sesuai dengan isi <i>email</i> yang ia kirim	105
5.18	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang pertama	105
5.19	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang kedua	106
5.20	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang ketiga	106
5.21	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang keempat	107
5.22	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang kelima	107
5.23	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang keenam	108
5.24	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang ketujuh	108
5.25	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang kedelapan	109
5.26	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang kesembilan	109
5.27	Diagram jawaban pertanyaan <i>System Usability Scale</i> yang kesepuluh	110
5.28	Diagram Skor System Usability Scale	110
B.1	Kuesioner bagian pertama	129
B.2	Kuesioner bagian Dosen pertanyaan pertama	130
B.3	Kuesioner bagian Dosen pertanyaan kedua dan ketiga	130
B.4	Kuesioner bagian Dosen pertanyaan keempat	131
B.5	Kuesioner bagian Mahasiswa pertanyaan pertama dan kedua	131
B.6	Kuesioner bagian Mahasiswa pertanyaan ketiga	132
B.7	Kuesioner bagian Mahasiswa pertanyaan keempat	132
B.8	Kuesioner bagian Mahasiswa pertanyaan kelima	132
B.9	Kuesioner bagian <i>System Usability Scale</i> dan Saran bagian 1	133
B.10	Kuesioner bagian <i>System Usability Scale</i> dan Saran bagian 2	134
B.11	Kuesioner bagian <i>System Usability Scale</i> dan Saran bagian 3	135

DAFTAR TABEL

2.1	Tabel perbandingan antar <i>stack</i>	9
4.1	Rincian method __construct	60
4.2	Rincian method checkEmail	60
4.3	Rincian method proceedEmail	61
4.4	Rincian method __construct	61
4.5	Rincian method proceedWebhook	62
4.6	Rincian method pushMessageToAllFollowers	62
4.7	Rincian method fifteenminutely	63
4.8	Rincian method __construct	64
4.9	Rincian method index	64
4.10	Rincian method read	64
4.11	Rincian method page	65
4.12	Rincian method pagination	65
4.13	Rincian method webhook	66
4.14	Rancangan Tabel Pengumuman	66
4.15	Rancangan Tabel PengumumanLineFollowers	67
5.1	Pengujian Filter <i>Email</i> Pengumuman	92
5.2	Pengujian Mengirim <i>Email</i> dengan Isi <i>Email</i> yang Variatif	92
5.3	Pengujian Notifikasi LINE	94
5.4	Tabel saran	111
B.1	Hasil Kuesioner Dosen	135
B.2	Hasil Kuesioner Mahasiswa	136
B.3	Hasil Kuesioner System Usability Scale (SUS)	137
B.4	Hasil Kuesioner Saran	139

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Pengumuman di jurusan Teknik Informatika UNPAR pada umumnya dilakukan lewat *email*. Pengumuman lewat *email* lebih praktis daripada pengumuman di papan pengumuman karena *email* dijamin sampai ke pihak yang dituju setelah dikirim. Namun, layanan *email* memiliki kotak masuk yang kurang terorganisir. Berbagai macam *email* yang masuk tercampur di kotak masuk sehingga dapat menyulitkan pemilik *email* untuk mencari *email* yang penting. Hal ini juga dapat mengakibatkan pengumuman-pengumuman penting tidak terbaca secara tidak sengaja.

Penelitian pada skripsi ini hendak membuat solusi untuk kekurangan *email* tersebut dengan cara membuat fitur kolektor pengumuman di BlueTape. BlueTape adalah aplikasi yang sudah ada yang berfungsi untuk membantu urusan-urusan *paper-based* di FTIS UNPAR menjadi *paperless*. Fitur ini akan mengumpulkan *email* yang berisi pengumuman dan menampilkannya di BlueTape. Agar penerima pengumuman tahu ada pengumuman baru di BlueTape, fitur ini juga akan memanfaatkan layanan LINE@. LINE@ adalah layanan dari *Line Corporation* yang memungkinkan pemilik bisnis atau organisasi membuat akun khusus (disebut akun LINE@) yang dapat mengirim pesan ke banyak pengikut secara bersamaan. Penerima pengumuman akan diminta untuk mengikuti akun LINE@ BlueTape sehingga dapat menerima notifikasi LINE apabila ada pengumuman baru. Agar BlueTape versi skripsi ini dapat diakses melalui internet, fitur ini juga akan memanfaatkan layanan Heroku. Heroku adalah layanan *cloud platform* pihak ketiga yang memungkinkan penggunaanya untuk membangun, menjalankan, dan mengoperasikan aplikasi pada *cloud*.

1.2 Rumusan Masalah

Rumusan masalah dari skripsi ini adalah:

- Bagaimana konsep dan cara kerja LINE@ dan Heroku?
- Bagaimana cara memodifikasi BlueTape agar dapat berjalan di Heroku?
- Bagaimana cara mengimplementasikan fitur kolektor pengumuman pada BlueTape?
- Bagaimana cara menguji fitur kolektor pengumuman pada BlueTape?

1.3 Tujuan

Tujuan dari skripsi ini adalah:

- Mempelajari konsep dan cara kerja LINE@ dan Heroku.
- Memodifikasi BlueTape agar dapat berjalan di Heroku.
- Mengimplementasikan fitur kolektor pengumuman pada BlueTape.
- Menguji fitur kolektor pengumuman pada BlueTape.

1.4 Batasan Masalah

Pada skripsi ini masalah dibatasi dengan batasan-batasan sebagai berikut:

- Fitur ini tidak akan menampilkan lampiran karena dapat membuat masalah lebih kompleks.
- Pengumuman di BlueTape dapat dilihat oleh semua mahasiswa dan dosen yang memiliki hak akses ke BlueTape.
- Semua akun yang mengikuti akun LINE@ BlueTape dapat menerima notifikasi apabila ada pengumuman baru.
- Sinkronisasi *email* dan pengiriman notifikasi LINE@ tidak dilakukan setiap saat, melainkan pada periode tertentu.

1.5 Metodologi

Metode penelitian pada skripsi ini sebagai berikut:

1. Melakukan studi literatur tentang PHP IMAP, LINE@, dan Heroku.
2. Menganalisis sistem BlueTape
3. Merancang sistem usulan
4. Memodifikasi BlueTape sehingga dapat mengumpulkan *email* yang berisi pengumuman.
5. Memodifikasi BlueTape sehingga dapat mengirim notifikasi ke akun Line@ BlueTape.
6. Memodifikasi BlueTape sehingga dapat berjalan di Heroku.
7. Melakukan pengujian.
8. Menulis dokumen skripsi.

1.6 Sistematika Pembahasan

Berikut adalah sistematika pembahasan skripsi ini :

1. Bab 1: Pendahuluan. Bab ini membahas gambaran umum dari skripsi ini. Bab ini berisi latar belakang, rumusan masalah, tujuan, batasan masalah, metode penelitian, dan sistematika penulisan.
2. Bab 2: Dasar Teori. Bab ini membahas dasar teori yang mendukung pembuatan skripsi ini, meliputi: Heroku, Cron, Gmail API, PHP IMAP, LINE, dan *System Usability Scale*.
3. Bab 3: Analisis. Bab ini membahas analisis sistem kini dan sistem usulan.
4. Bab 4: Perancangan. Bab ini membahas perancangan fitur kolektor pengumuman, meliputi: perancangan kelas, perancangan basis data, dan perancangan antarmuka.
5. Bab 5: Implementasi dan Pengujian. Bab ini membahas implementasi dan pengujian fitur kolektor pengumuman pada BlueTape.
6. Bab 6: Kesimpulan dan saran. Bab ini berisi kesimpulan yang didapat dari pembangunan fitur kolektor pengumuman pada BlueTape dan saran oleh peneliti kepada pembaca yang hendak melanjutkan penelitian ini.

BAB 2

LANDASAN TEORI

Bab ini berisi landasan teori yang dipakai pada skripsi ini.

2.1 Heroku [1]

Heroku adalah *cloud platform* yang memungkinkan penggunanya untuk membangun, menjalankan, dan mengoperasikan aplikasi pada *cloud*. Heroku mendukung beberapa bahasa pemrograman, meliputi: Ruby, Node.js, Java, Python, Clojure, Scala, Go, dan PHP.

2.1.1 Arsitektur Heroku

Heroku memungkinkan penggunanya untuk menyebarkan, menjalankan, dan mengelola aplikasi yang ditulis di dalam bahasa yang didukung oleh Heroku. Heroku mendefinisikan aplikasi sebagai gabungan dari *source code* (kode program) yang ditulis di dalam salah satu bahasa yang didukung Heroku, deskripsi *dependency* yang dipakai, dan *file Procfile*.

2.1.1.1 *Dependency*

Agar aplikasi dapat dibangun dan dijalankan, aplikasi mungkin membutuhkan kode program dari pihak ketiga. Kode program tersebut disebut *dependency*. Aturan peletakan deskripsi *dependency* berbeda-beda untuk tiap bahasa. Contoh: Aplikasi dengan bahasa Node.js menuliskan deskripsi *dependency* pada *file package.json*.

2.1.1.2 *Dyno*

Dyno adalah wadah aplikasi berbasis Unix yang menjalankan kode program di Heroku. Pada umumnya, Heroku akan menjalankan satu *web dyno* secara otomatis saat aplikasi disebar ke Heroku untuk pertama kali.

Dyno dikelompokkan ke dalam tiga kelompok, yaitu:

- *Web dyno*

Web dyno adalah *dyno* yang berjalan pada *process type web*. *Web dyno* adalah satu-satunya *dyno* yang dapat menerima arus HTTP dari *router* Heroku.

- *Worker dyno*

Worker dyno merujuk kepada *dyno* yang berjalan pada *process type* yang disebutkan di dalam *Procfile* selain *process type web*. *Worker dyno* biasanya digunakan untuk pekerjaan pada latar belakang, sistem antrian, dan pekerjaan berjangka waktu tertentu.

- *One-off dyno*

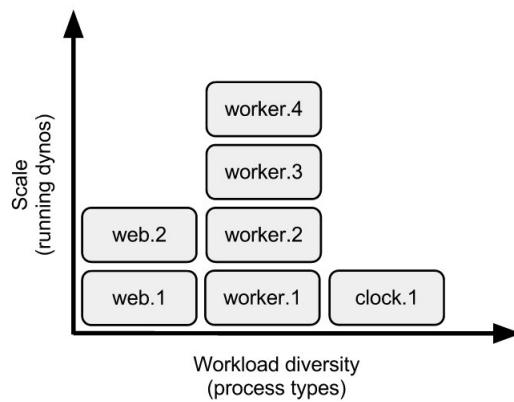
One-off dyno adalah *dyno* yang bersifat sementara. *One-off dyno* tidak dituliskan pada *Procfile* untuk menjalankannya, melainkan menggunakan *command shell*. Contoh penggunaan *one-off dyno* adalah pada saat migrasi basis data.

Heroku menyediakan berbagai macam tipe *dyno*, yaitu: tipe *free*, tipe *hobby*, tipe *standard*, dan tipe *performance*. Tipe *dyno* mempengaruhi karakteristik *dyno* dan kinerja aplikasi. Pada aplikasi dengan tipe *dyno free* dan tipe *dyno hobby*, semua *process type* pada aplikasi tersebut hanya dapat memakai tipe *dyno* tersebut. Namun, tipe *standard* dan tipe *dyno performance* dapat digunakan di aplikasi yang sama. Pada tipe *dyno free* dan tipe *hobby*, jumlah *dyno* yang dapat dipakai untuk tiap *process type* adalah satu. Pada tipe *dyno performance*, jumlah *dyno* yang dapat dipakai untuk tiap *process type* adalah sepuluh. Pada tipe *dyno free*, jumlah *dyno* yang dapat dijalankan secara bersamaan pada satu aplikasi adalah dua. Pada tipe *dyno* lainnya, jumlah *dyno* yang dapat dijalankan secara bersamaan pada satu aplikasi adalah seratus.

Tipe *dyno free* memiliki beberapa batasan, yaitu dapat masuk ke kondisi *sleep* dan penggunaan per bulannya dibatasi dengan *free dyno hours*. Kondisi *sleep* adalah kondisi saat *web dyno* tidak aktif. Kondisi *sleep* terjadi apabila tidak ada arus HTTP masuk setelah 30 menit aktif. Apabila ada arus HTTP masuk pada saat aplikasi tertidur, maka aplikasi akan aktif kembali. Namun, ada keterlambatan saat memproses arus HTTP tersebut. *Free dyno hours* adalah jumlah jam saat *web dyno* tidak dalam kondisi *sleep*, saat *worker dyno* berjalan, dan saat *one-off dyno* berjalan. Akun yang belum diverifikasi dapat menggunakan 550 *free dyno hours* tiap bulannya, sedangkan akun yang telah diverifikasi dapat menggunakan 1000 *free dyno hours* tiap bulannya. Setiap aplikasi dengan tipe *dyno free* milik seorang pengguna akan memakai kuota *free dyno hours* yang sama. Apabila penggunaan *dyno* telah melebihi 80% kuota, maka pengguna akan dikirim *email* peringatan. Apabila penggunaan telah melebihi 100% kuota, maka pengguna akan dikirim *email* notifikasi kedua dan aplikasinya akan masuk ke kondisi *sleep* selama sisa hari pada bulan tersebut. Jumlah *free dyno hours* yang tersisa dapat dilihat dengan mengetikkan "`heroku ps -a`" diikuti dengan nama aplikasi di *command shell* atau dengan melihat bagian *free dyno usage* pada halaman billing (<https://dashboard.heroku.com/account/billing>).

2.1.1.3 Process Type

Process type adalah prototipe yang menjadi tempat dimana *dyno* dibentuk. *Process type* yang tersedia dapat berbeda untuk tiap bahasa pemrograman. Namun, ada dua *process type* spesial pada Heroku, yaitu *process type web* dan *process type release*. *Process type web* adalah satu-satunya *process type* yang dapat menerima arus HTTP eksternal dari *router* Heroku. *Process type release* adalah *process type* yang digunakan untuk menyebutkan perintah yang dijalankan selama fase *release*.



Gambar 2.1: Diagram hubungan antara *process type* dan *dyno*

Process type dan *dyno* saling berhubungan. Hubungan tersebut dapat dilihat pada Gambar 2.1. Sumbu x menyatakan *process type* yang dipakai, sementara sumbu y menyatakan jumlah *dyno* yang berjalan pada *process type* tersebut. Semakin banyak *dyno* pada suatu *process type* maka konkurensi untuk pekerjaan yang ditangani *process type* tersebut akan meningkat. Semakin banyak *process*

type maka semakin beragam beban kerja.

Jumlah *dyno* yang berjalan pada *process type* dapat diatur dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku ps:scale <process=dyno list>
```

Keterangan:

- <process=dyno list>: daftar pasangan *process type* dengan jumlah *dyno* yang ditugaskan untuk proses tersebut.

Contoh:

```
$ heroku ps:scale web=2 worker=4 clock=1
```

2.1.1.4 Procfile

Heroku dapat mengetahui bagian aplikasi mana yang dapat dijalankan untuk sebagian besar bahasa pemrograman. Contoh: Heroku langsung mengetahui bagian "main" pada file `package.json` adalah bagian aplikasi yang dapat dijalankan pada aplikasi dengan bahasa Node.js. Namun, Heroku terkadang membutuhkan petunjuk secara eksplisit untuk mengetahui bagian yang dapat dijalankan. Petunjuk tersebut ditulis di sebuah file teks bernama Procfile.

Procfile adalah file tanpa ekstensi file (Contoh ekstensi file: ".txt") yang diletakkan di direktori root. Isi dari Procfile adalah teks yang tiap barisnya memiliki format sebagai berikut:

```
<process type>: <command>
```

Keterangan:

- <process type>: nama *process type*, contoh: `urgentworker`. Aplikasi yang sederhana hanya membutuhkan *process type* `web` saja.
- <command>: perintah yang harus dijalankan oleh setiap *dyno* dari *process type* tersebut pada saat startup.

2.1.1.5 Config vars

Pada Heroku, *environment variable* disimpan di dalam *config vars*. *Environment variable* adalah konfigurasi yang dapat berubah-ubah pada lingkungan yang berbeda. Konfigurasi tersebut meliputi informasi *database*, informasi kredensial, atau informasi lain yang bersifat spesifik pada aplikasi. Penggunaan *environment variable* dapat menghindari tersimpannya informasi penting di pengontrol versi.

Config vars dapat diatur dengan tiga cara :

- Menggunakan Heroku CLI

Heroku CLI menggunakan *command shell* untuk menerima perintah yang harus dijalankan. Berikut daftar perintah yang berkaitan dengan *config vars* beserta fungsinya:

- Menampilkan seluruh *config vars* beserta nilainya :

```
$ heroku config
```

- Menampilkan nilai dari *config vars* tertentu

```
$ heroku config : get <config vars>
```

Keterangan :

- * <config vars>: nama config vars
- Menambah config vars

```
$ heroku config : set <config vars> = <config values>
```

Keterangan :

- * <config vars>: nama config vars
- * <config values>: nilai dari config vars tersebut

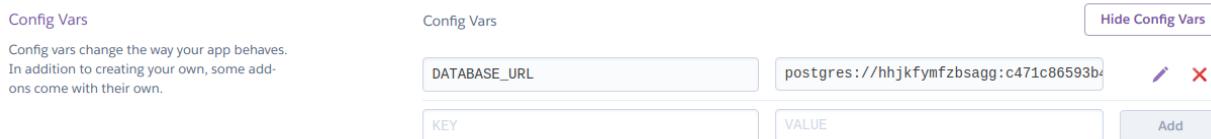
- Menghapus config vars

```
$ heroku config : unset <config vars>
```

Keterangan :

- * <config vars> : nama config vars

- Menggunakan Heroku dashboard



Gambar 2.2: Config vars pada dashboard Heroku

Heroku *dashboard* berada di <https://dashboard.heroku.com/>. Setelah memilih menu "Settings" di aplikasi yang dipilih, pilih menu "Config vars" untuk melihat, menambah, dan menghapus config vars. Tampilan menu "Config vars" dapat dilihat pada Gambar 2.2.

- Menggunakan Heroku Platform API

Config vars dapat diatur dengan mengirimkan PATCH *request* ke Heroku Platform API dengan format berikut:

```
PATCH /apps/{app_id_or_name}/config-vars
```

Contoh *request* menggunakan curl:

```
$ curl -n -X PATCH https://api.heroku.com/apps/$APP_ID_OR_NAME/config-vars \
-d '{
  "FOO": "bar",
  "BAZ": "qux"
}' \
-H "Content-Type: application/json" \
-H "Accept: application/vnd.heroku+json; version=3"
```

Contoh *response* dari Heroku Platform API:

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2012 12:00:00 GMT
RateLimit-Remaining: 4500
{
  "FOO": "bar",
  "BAZ": "qux"
}
```

Apabila pengguna ingin melihat *config vars*, maka pengguna dapat mengirimkan GET *request* dengan format berikut:

```
GET /apps/{app_id_or_name}/config-vars
```

Contoh *request* menggunakan curl:

```
$ curl -n https://api.heroku.com/apps/$APP_ID_OR_NAME/config-vars \
-H "Accept: application/vnd.heroku+json; version=3"
```

Contoh *response* dari Heroku Platform API:

```
HTTP/1.1 200 OK
ETag: "0123456789abcdef0123456789abcdef"
Last-Modified: Sun, 01 Jan 2012 12:00:00 GMT
RateLimit-Remaining: 4500
{
  "FOO": "bar",
  "BAZ": "qux"
}
```

Dalam mengatur *config vars*, ada beberapa hal yang harus diperhatikan:

- Setiap *config vars* ditambah atau dihapus, aplikasi akan dimulai ulang dan *release* baru akan dibuat.
- Jika aplikasi menggunakan *add-ons*, biasanya *add-ons* tersebut akan menambahkan satu atau lebih *config vars* ke aplikasi. Nilai dari *config vars* tersebut mungkin diperbarui oleh penyedia *add-ons* kapan saja.
- Data *config vars* (kombinasi dari semua kunci dan nilainya) tidak dapat melebihi 32 kb per aplikasi
- Nama *config vars* tidak boleh diawali dengan tanda hubung bawah dua kali (__).
- Nama *config vars* tidak bisa diawali dengan HEROKU_, kecuali ditambahkan oleh *platform* Heroku sendiri.

2.1.1.6 Add-ons

Pengguna Heroku dapat memanfaatkan *add-ons* untuk memakai layanan penyokong, misalnya basis data, sistem antrean, layanan *email*, dan lain-lain. Pengguna Heroku dapat mencari *add-ons* di situs web *Elements Marketplace* (<https://elements.heroku.com/addons>). Beberapa *add-ons* di dalam situs web tersebut disediakan oleh Heroku dan beberapa lainnya disediakan oleh pihak ketiga. Pengguna Heroku dapat memasang *add-ons* pada aplikasi dengan menekan tombol "Install" di situs web *Elements Marketplace* atau dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku addons:create <nama add-ons>:<tipe add-ons>
```

Keterangan:

- <nama add-ons> : nama *add-ons*
- <tipe add-ons> : tipe *add-ons*

Contoh:

```
$ heroku addons:create heroku-redis:hobby-dev
```

Menambah *add-ons* selain *add-ons* Heroku Postgres dan Heroku Connect membutuhkan verifikasi akun.

2.1.1.7 Buildpack

Buildpack	Shorthand
Ruby	heroku/ruby
Node.js	heroku/nodejs
Clojure	heroku/clojure
Python	heroku/python
Java	heroku/java
Gradle	heroku/gradle
Grails 3.x	heroku/gradle
Scala	heroku/scala
Play 2.x	heroku/scala
PHP	heroku/php
Go	heroku/go

Gambar 2.3: Tabel buildpack heroku

Buildpack adalah kumpulan kode program yang bertugas untuk mengubah *source code* menjadi *slug*, sehingga *dyno* dapat mengeksekusinya. Heroku biasanya akan secara otomatis menugaskan *buildpack* yang akan dipakai setelah *build* pertama berhasil. Daftar *buildpack* yang tersedia di Heroku dapat dilihat pada Gambar 2.3. Kolom *Buildpack* menyatakan nama *buildpack* dan kolom *shorthand* menyatakan nama panggil *buildpack* di Heroku CLI.

Jumlah *buildpack* yang dipakai oleh satu aplikasi biasanya hanya satu, tapi ada beberapa kasus *buildpack* yang dipakai lebih dari satu. Contoh kasus *buildpack* tambahan dibutuhkan:

- Satu aplikasi memiliki lebih dari satu bahasa pemrograman.
- Aplikasi menjalankan *daemon*.
- Aplikasi membutuhkan *dependency* yang diambil dengan *APT (Advanced Package Tool)*.

Buildpack dapat diatur dengan mengetikkan perintah di *command shell* menggunakan Heroku CLI. Berikut daftar perintah yang berkaitan dengan *buildpack* beserta fungsinya:

- Mengatur *buildpack* yang dipakai saat aplikasi pertama kali dibuat:

```
$ heroku create myapp --buildpack <nama buildpack>
```

Keterangan:

- <nama buildpack>: nama panggil *buildpack*, contoh: `heroku/php`.

- Mengubah dengan mengatur nilai *buildpack*:

```
$ heroku buildpacks:set <nama buildpack>
```

Keterangan:

- <nama buildpack>: nama panggil *buildpack*, contoh: `heroku/php`.

- Menghilangkan *buildpack* dari aplikasi:

```
$ heroku buildpacks:remove <nama buildpack>
```

Keterangan:

- <nama buildpack>: nama panggil *buildpack*, contoh: heroku/php.
- Mencari buildpack:

```
$ heroku buildpacks:search <kata kunci>
```

Keterangan:

- <kata kunci>: kata kunci pencarian, misalnya nama bahasa pemrograman. Contoh kata kunci: elixir.
- Menampilkan informasi *buildpack*:

```
$ heroku buildpacks:info <nama buildpack>
```

Keterangan:

- <nama buildpack>: nama panggil *buildpack*, contoh: heroku/php.
- Mengembalikan aplikasi ke *buildpack* awalnya:

```
$ heroku buildpacks:clear
```

- Mengatur urutan eksekusi *buildpack*:

```
$ heroku buildpacks:set --index <index> <nama buildpack>
```

Keterangan:

- <index>: urutan eksekusi *buildpack*
- <nama buildpack>: nama panggil *buildpack*, contoh : heroku/php.
- Melihat daftar *buildpack* yang dipakai pada satu aplikasi:

```
$ heroku buildpacks
```

2.1.1.8 Stack

Stack merujuk pada sistem operasi yang dikelola dan dipelihara oleh Heroku. *Stack* biasanya memakai sistem operasi Linux, misalnya Ubuntu. Saat skripsi ini dibuat, Heroku menyediakan tiga pilihan *stack*: Cedar-14, Heroku-16, dan Heroku-18. Perbandingan ketiga *stack* tersebut dapat dilihat di Tabel 2.1.

Tabel 2.1: Tabel perbandingan antar *stack*

Nama <i>stack</i>	Sistem Operasi yang dipakai	Periode dukungan
Cedar-14	Ubuntu 14.04	Sampai bulan April 2019
Heroku-16	Ubuntu 16.04	Sampai bulan April 2021
Heroku-18	Ubuntu 18.04	Sampai bulan April 2023

Pengguna Heroku dapat melihat *stack* yang dipakai pada suatu aplikasi dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku stack
```

Pada saat aplikasi baru dibentuk, Heroku akan secara otomatis memakai *stack* yang memiliki periode dukungan paling panjang. Namun, pengguna dapat mengganti *stack* yang dipakai dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku stack:set <stack>
```

Keterangan:

- <*stack*>: nama *stack*, contoh: cedar-14.

2.1.1.9 Region

Aplikasi yang ada di Heroku dapat disebarluaskan ke lokasi geografis yang berbeda. Pada heroku, lokasi penyebaran aplikasi disebut *region*. *Region* yang tersedia untuk pengguna Heroku yang merupakan perorangan adalah lokasi dengan *runtime Common Runtime*, sedangkan *region* yang tersedia untuk pengguna Heroku yang merupakan badan bisnis (Heroku Enterprise) adalah lokasi dengan *runtime Private Space*. Pada saat skripsi ini dibuat, pengguna Heroku yang merupakan perorangan hanya dapat memilih antara *Europe* (Eropa) dan *United States* (Amerika Serikat). Apabila pengguna tidak memilih *region* saat membuat aplikasi, maka Heroku akan secara otomatis mengatur *region* menjadi *United States*. Beberapa layanan yang disediakan untuk membangun aplikasi di Heroku hanya dapat digunakan di *region* tertentu, sehingga pengguna lebih baik merencanakannya terlebih dahulu sebelum memilih *region*.

Berikut perintah-perintah penting yang berkaitan dengan *region* yang dapat diketikkan pada *command shell* :

- Memeriksa *region* yang tersedia di Heroku:

```
$ heroku regions
```

- Mengatur *region* aplikasi:

```
$ heroku create --region <id region>
```

Keterangan:

- <*id region*>: *id region* yang ingin dipakai, contoh: eu. *Id region* bisa dilihat dengan memeriksa daftar region yang tersedia.

- Memeriksa *region* yang dipakai oleh aplikasi:

```
$ heroku info
```

2.1.1.10 Releases

Setiap pengguna melakukan *deploy*, merubah *config vars*, dan merubah daftar *add-ons* yang dipakai, Heroku akan mencatat aktivitas tersebut di *releases*. *Releases* adalah buku besar yang mencatat setiap ada perubahan baru pada aplikasi. *Releases* berbentuk tabel yang memiliki empat kolom, yaitu "Rel", "Change", "By", dan "When". Kolom "rel" berisi versi aplikasi, kolom "Change" berisi keterangan singkat perubahan yang terjadi, kolom "By" berisi alamat *email* pengguna yang melakukan perubahan, dan kolom "When" berisi waktu perubahan tersebut terjadi. Pengguna dapat melihat *releases* dengan mengetikkan perintah berikut di *command shell*:

```
$ heroku releases
```

Berikut contoh isi *releases*:

```
Rel Change By When
-----
v52 Config add AWS_S3_KEY jim@example.com 5 minutes ago
v51 Deploy de63889 stephan@example.com 7 minutes ago
v50 Deploy 7c35f77 stephan@example.com 3 hours ago
v49 Rollback to v46 joe@example.com 2010-09-12 15:32:17 -0700
```

Releases berguna ketika pengguna ingin mengembalikan aplikasi ke versi tertentu. Cara mengembalikan aplikasi ke versi tertentu adalah dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku releases:rollback <version>
```

Keterangan:

- <version>: Versi aplikasi

2.1.2 Log

Log adalah catatan setiap proses yang terjadi di aplikasi. Heroku menggunakan Logplex untuk menyampaikan *log* ini. Logplex akan secara otomatis menambahkan entri *log* baru dari semua *dyno* yang berjalan di aplikasi, dan juga komponen lain seperti *router*. Pengguna dapat melihat *log* dengan cara mengetikkan perintah berikut pada *command shell*:

```
$ heroku logs
```

Isi *log* memiliki format sebagai berikut:

```
<timestamp> <source>[<dyno>]: <message>
```

Keterangan:

- <timestamp>: Waktu *log* dicatat dengan tingkat ketelitian hingga mikrodetik.
- <source>: Jika *log* berasal dari *dyno*, maka isinya adalah "app". Jika *log* berasal dari komponen sistem Heroku (HTTP *router*, *dyno manager*) maka isinya adalah "heroku".
- dyno: Nama *dyno* atau komponen yang dicatat di *log*.
- message: Keterangan *log*.

Berikut contoh isi *log*:

```
2010-09-16T15:13:46.677020+00:00 app[web.1]: Processing PostController#list (for
  ↵ 208.39.138.12 at 2010-09-16 15:13:46) [GET]
2010-09-16T15:13:46.677023+00:00 app[web.1]: Rendering template within layouts/
  ↵ application
2010-09-16T15:13:46.677902+00:00 app[web.1]: Rendering post/list
2010-09-16T15:13:46.678990+00:00 app[web.1]: Rendered includes/_header (0.1ms)
2010-09-16T15:13:46.698234+00:00 app[web.1]: Completed in 74ms (View: 31, DB: 40)
  ↵ | 200 OK [http://myapp.herokuapp.com/]
2010-09-16T15:13:46.723498+00:00 heroku[router]: at=info method=GET path="/posts"
  ↵ host=myapp.herokuapp.com" fwd="204.204.204.204" dyno=web.1 connect=1ms
  ↵ service=18ms status=200 bytes=975
2010-09-16T15:13:47.893472+00:00 app[worker.1]: 2 jobs processed at 16.6761 j/s,
  ↵ 0 failed ...
```

Heroku mengelompokkan *log* ke dalam empat kelompok:

- *App logs*

Log yang berasal dari aplikasi, termasuk *log* yang dihasilkan oleh kode program dan *dependency*. Pengguna dapat melihat *app logs* dengan cara mengetikkan perintah berikut pada *command shell*:

```
$ heroku logs --source app
```

- *System Logs*

Log yang berisi pesan tentang tindakan yang diambil oleh infrastruktur Heroku mewakili aplikasi, misalnya: saat terjadi *error*. Pengguna dapat melihat *system logs* dengan cara mengetikkan perintah berikut pada *command shell*:

```
$ heroku logs --source heroku
```

- *API logs*

Log yang berisi pesan tentang tindakan administratif yang diambil oleh pembuat aplikasi, misalnya: saat melakukan *deploy*. Pengguna dapat melihat *API logs* dengan cara mengetikkan perintah berikut pada *command shell*:

```
$ heroku logs --source app --dyno api
```

- *Add-on logs*

Log yang berisi pesan dari layanan *add-ons*.

Pengguna Heroku dapat menuliskan *log* dengan cara megirimkan pesan ke *standard out (stdout)* atau *standard error (stderr)*. Cara mengirimkannya berbeda-beda tergantung bahasa pemrograman yang dipakai. Pada bahasa PHP, *log* dapat dikirim dengan menggunakan fungsi "*error_log*" atau menulis ke "*php://stderr*".

Contoh penggunaan fungsi "*error_log*":

```
error_log("hello, this is a test!");
```

Contoh menulis ke "*php://stderr*":

```
file_put_contents("php://stderr", "hello, this is a test!\n");
```

Apabila aplikasi menggunakan *framework* Codeigniter, maka pembuat aplikasi perlu menambahkan file *application/core/MY_Log.php* dan mengatur "sub-class prefix" menjadi "MY_". Isi file tersebut adalah sebagai berikut:

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

// this class is adapted from system/libraries/Log.php
/**
 * CodeIgniter
 *
 * An open source application development framework for PHP 5.1.6 or newer
 *
 * @package CodeIgniter
 * @author EllisLab Dev Team
 * @copyright Copyright (c) 2008 - 2014, EllisLab, Inc.
```

```
* @copyright Copyright (c) 2014 - 2015, British Columbia Institute of Technology
  ↪ (http://bcit.ca/)
* @license http://codeigniter.com/user_guide/license.html
* @link http://codeigniter.com
* @since Version 1.0
* @filesource
*/
// -----
/***
 * Logging Class
 *
 * @package CodeIgniter
 * @subpackage Libraries
 * @category Logging
 * @author EllisLab Dev Team
 * @link http://codeigniter.com/user_guide/general/errors.html
 */
class MY_Log {

    protected $_threshold = 1;
    protected $_date_fmt = 'Y-m-d H:i:s';
    protected $_levels = array('ERROR' => '1', 'DEBUG' => '2', 'INFO' => '3', 'ALL'
      ↪ ' => '4');

    /**
     * Constructor
     */
    public function __construct()
    {
        $config =& get_config();

        if (is_numeric($config['log_threshold']))
        {
            $this->_threshold = $config['log_threshold'];
        }

        if ($config['log_date_format'] != '')
        {
            $this->_date_fmt = $config['log_date_format'];
        }
    }

    // -----
    /**
     * Write Log to php://stderr
     *
     * Generally this function will be called using the global log_message()
     ↪ function
}
```

```

/*
 * @param string the error level
 * @param string the error message
 * @param bool whether the error is a native PHP error
 * @return bool
 */
public function write_log($level = 'error', $msg, $php_error = FALSE)
{
    $level = strtoupper($level);

    if ( ! isset($this->_levels[$level]) OR ($this->_levels[$level] > $this->
        _threshold))
    {
        return FALSE;
    }

    file_put_contents('php://stderr', $level.' '.(($level == 'INFO') ? ' - ' :
        '-').' '.date($this->_date_fmt). ' --> '.$msg."\n");

    return TRUE;
}

// END Log Class

/* End of file MY_Log.php */
/* Location: ./application/core/MY_Log.php */

```

2.1.3 Deploy Perangkat Lunak

Heroku menggunakan Git sebagai sarana utama untuk melakukan *deploy* ke Heroku. Deploy adalah proses penyebaran aplikasi dari satu lingkungan ke lingkungan lain, misalnya dari lingkungan lokal aplikasi ke lingkungan Heroku. Namun, Heroku juga menyediakan cara lain untuk melakukan *deploy*:

- Docker
- GitHub
- Tombol Deploy di dashboard Heroku
- WAR deployment

2.1.3.1 Deploy Menggunakan Git

Deploy dengan Git dapat dilakukan apabila Git telah terpasang. Git dapat dipasang dengan mengikuti petunjuk unduhan pada <https://git-scm.com>. Sebelum *deploy* dengan Git dapat dilakukan, Git perlu diinisialisasi terlebih dahulu dengan mengetikkan perintah berikut pada *command shell* :

```
$ git init
```

Setelah melakukan inisialisasi Git, aplikasi Heroku dapat dibuat. Ketika aplikasi Heroku dibuat, maka *git remote* secara otomatis juga dibuat. Nama *remote* dapat diperiksa dengan mengetikkan perintah berikut pada *command shell* :

```
$ git remote
```

Nama *remote* dapat diubah dengan mengetikkan perintah berikut pada *command shell* :

```
$ git remote rename <nama lama> <nama baru>
```

Keterangan:

- <nama lama>: nama *remote* yang ingin diganti.
- <nama baru>: nama baru untuk *remote* tersebut.

Deploy dapat dimulai dengan mengetikkan perintah berikut pada *command shell* :

```
$ git add <nama file>
$ git commit -m "<message>"
$ git push <nama remote> <nama branch>
```

Keterangan :

- <nama file>: nama *file* yang telah diubah. Menulis tanda titik untuk nama *file* mewakili semua *file* yang telah berubah.
- <message>: pesan yang menerangkan hal yang ingin disebar.
- <nama remote> : nama *remote* dari tujuan deploy. Bila *developer* tidak mengubah nama *remote*, nama remotenya adalah **heroku**.
- <nama branch> : nama cabang dari tujuan deploy. Heroku secara otomatis membuat satu cabang bernama **master**.

2.1.3.2 Deploy Menggunakan Docker

Deploy dengan Docker dapat dilakukan apabila Docker telah terpasang dan pengguna Heroku telah masuk ke akun Heroku. Langkah-langkah untuk melakukan *deploy* menggunakan Docker adalah sebagai berikut:

- Masuk ke *Container Registry* dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku container:login
```

- *Clone source code* contoh dari Alpine dengan mengetikkan perintah berikut pada *command shell*:

```
$ git clone https://github.com/heroku/alpinehelloworld.git
```

- Membuat aplikasi Heroku baru dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku create
```

- Membangun *image* dan melakukan *deploy* ke *Container Registry* dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku container:push web
```

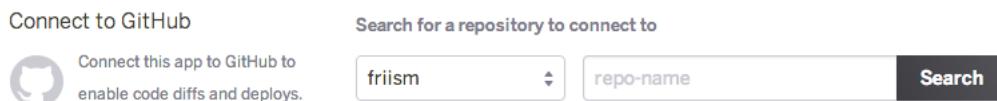
- Melepaskan *image* ke aplikasi dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku container:release web
```

- Membuka aplikasi dengan mengetikkan perintah berikut pada *command shell*:

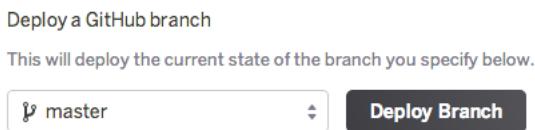
```
$ heroku open
```

2.1.3.3 Deploy Menggunakan GitHub

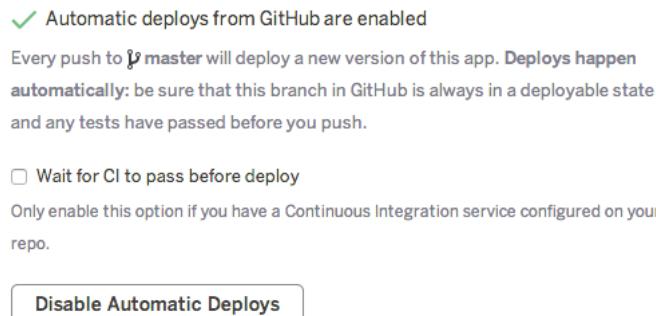


Gambar 2.4: *Deploy* menggunakan Github *Dashboard*

Deploy dengan cara ini membuat Heroku melakukan *deploy* secara otomatis ke GitHub apabila *build* berhasil. *Deploy* ini dapat dilakukan apabila GitHub *integration* telah aktif dan akun GitHub telah terautentikasi pada Heroku. Autentikasi ini hanya perlu dilakukan satu kali per satu akun Heroku. Setelah kedua hal tersebut telah siap, sambungkan *repository* GitHub dengan aplikasi Heroku. (Gambar 2.4).



Gambar 2.5: *Deploy* menggunakan Github secara manual



Gambar 2.6: *Deploy* menggunakan Github secara otomatis

Ada dua cara untuk melakukan *deploy*, yaitu secara manual dan secara otomatis. Untuk cara manual, *deploy* dilakukan dari GitHub (Gambar 2.5). Untuk cara otomatis, *deploy* dilakukan dengan mengaktifkan "Automatic deploys from GitHub" (Gambar 2.6).

2.1.3.4 Deploy Langsung di situs web Heroku

Pada situs web Heroku, tombol "*Deploy to Heroku*" memungkinkan pengguna untuk melakukan *deploy* aplikasi tanpa meninggalkan situs web Heroku. Cara ini hampir tidak memerlukan konfigurasi.

Sebelum dapat melakukan *deploy* dengan cara ini, aplikasi harus memiliki *file app.json* yang sah pada direktori **root**, dan *source code* aplikasi harus berada di *repository* GitHub.

app.json adalah *file* berisi deskripsi aplikasi *web*. Isinya dapat berupa *environment variable*, *add-ons*, dan informasi lain yang diperlukan untuk menjalankan aplikasi pada Heroku. Heroku tidak mewajibkan penulisan informasi tertentu, tapi Heroku merekomendasikan untuk setidaknya menuliskan nama aplikasi(**name**), deskripsi aplikasi (**description**), dan logo aplikasi (**logo**). Berikut contoh isi dari *app.json* :

```
{
  "name": "Node.js Sample",
  "description": "A barebones Node.js app using Express 4",
  "repository": "https://github.com/heroku/node-js-sample",
  "logo": "https://node-js-sample.herokuapp.com/node.png",
  "keywords": ["node", "express", "static"]
}
```

2.1.3.5 WAR Deployment

WAR (*Web Application Archive*) adalah jenis *file* arsip yang digunakan untuk membungkus aplikasi *web*. Dokumen ini dapat berisi halaman *web* statis, *file XML*, dan lain-lain. [7]

WAR deployment dapat dilakukan menggunakan *plugin* Heroku Java CLI Plugin. Pemasangan dan pembaruan *plugin* ini dapat dilakukan dengan mengetikkan perintah berikut pada *command shell*:

```
heroku plugins:install java
```

Deploy dapat dilakukan dengan mengetikkan perintah berikut pada *command shell*:

```
heroku war:deploy <path_to_war_file> --app <app_name>
```

Keterangan:

- <path_to_war_file>: letak *file* WAR.
- <app_name>: nama aplikasi.

2.1.4 Ephemeral Filesystem

Heroku memiliki *ephemeral filesystem*, yang berarti perubahan apapun pada *file* di lingkungan Heroku tidak akan disimpan secara permanen. Kondisi *file* di Heroku akan selalu dikembalikan ke *deploy baru*. Perubahan *file* hanya akan bertahan paling lama satu hari. Kondisi ini membuat aplikasi di Heroku membutuhkan layanan dari pihak ketiga untuk menyimpan hasil unggahan pengguna aplikasi.

2.1.5 Basis Data dan Manajemen Data

Heroku menyediakan tiga layanan data untuk semua pelanggan :

- Heroku Postgres
- Heroku Redis
- Apache Kafka

Heroku juga menyediakan pilihan lain untuk pengguna Heroku Enterprise, yaitu Heroku Connect. Selain itu, Heroku juga memungkinkan penggunaan layanan data dari pihak ketiga. Layanan data dari pihak ketiga ini tersedia sebagai add-ons.

2.1.5.1 Heroku Postgres

Heroku Postgres adalah basis data SQL yang disediakan secara langsung oleh Heroku. Heroku Postgres dapat diakses oleh bahasa apapun dengan PostgreSQL *driver*. Heroku secara otomatis membuat satu basis data menggunakan *add-ons* Heroku Postgres setiap aplikasi dibuat. Perintah berikut dapat diketikkan pada *command shell* apabila ingin menambah basis data baru:

```
$ heroku addons:create heroku-postgresql:<PLAN_NAME>
```

Keterangan :

- <PLAN_NAME> : nama *plan* Heroku Postgres yang ingin dipakai. Heroku secara otomatis menggunakan Heroku Postgres tipe **hobby-dev**.

Heroku menawarkan lima *plan* untuk Heroku Postgres, yaitu:

- Hobby Tier: *plan* dengan toleransi gagal bekerja sampai 4 jam per bulan.
- Standard Tier: *plan* dengan toleransi gagal bekerja sampai 1 jam per bulan.
- Premium Tier: *plan* dengan toleransi gagal bekerja sampai 15 menit per bulan.
- Private Tier: *plan* yang dapat dipilih oleh pengguna Heroku *Enterprise*, memiliki toleransi gagal bekerja sampai 15 menit per bulan.
- Shield Tier : *plan* yang dapat dipilih oleh pengguna Heroku *Enterprise*, memiliki toleransi gagal bekerja sampai 15 menit per bulan. Kelebihannya daripada *Private Tier* adalah basis data ini memiliki keamanan yang lebih ketat.

Gambar 2.7 menunjukkan tabel perbandingan antar *plan*. Di antara lima *plan* Heroku Postgres, hanya *plan Hobby* yang gratis. *Plan* lain memiliki harga yang bervariasi berdasarkan ukuran RAM, batas penyimpanan, dan batas koneksi yang bisa dibuat.

Heroku Postgres tier	Downtime Tolerance	Fork	Follow	Rollback	HA	Disk Encryption
Hobby	< 4 hr downtime per mo.	No	No	No	No	No
Standard	< 1 hr downtime per mo.	Yes	Yes	4 days	No	Yes
Premium	< 15 min downtime per mo.	Yes	Yes	1 week	Yes	Yes
Private	< 15 min downtime per mo.	Yes	Yes	1 week	Yes	Yes
Shield	< 15 min downtime per mo.	Yes	Yes	1 week	Yes	Yes

Gambar 2.7: Tabel plan Heroku Postgres

Semua *plan* memiliki fitur yang sama:

- Dikelola secara otomatis oleh Heroku.
- Data dijamin tidak akan hilang.
- Dapat melakukan *backup* basis data secara harian menggunakan Heroku PGBackups.
- Dapat menggunakan layanan Dataclips (<https://data.heroku.com/dataclips>) untuk berbagi data dan *query*.
- Akses basis data diproteksi dengan SSL.
- Menjalankan Postgres versi 9.4, 9.5, 9.6, atau 10 tanpa modifikasi
- Tersedia ekstensi Postgres

- Memiliki tampilan *web* yang dapat diakses pada situs web <https://data.heroku.com/>

Developer juga dapat menambahkan versi yang ingin dipakai dengan cara menambahkan `-version` di belakang perintah tersebut, contoh :

```
$ heroku addons:create heroku-postgresql:<PLAN_NAME>--version=9.5
```

Secara otomatis, Heroku menggunakan versi paling baru dari Heroku Postgres. Saat skripsi ini ditulis, versi terbaru adalah versi 10.

Setelah dipasang, Heroku akan secara otomatis menambahkan *config vars* `DATABASE_URL` ke aplikasi. Apabila Heroku Postgres yang dipakai ada lebih dari satu, nama *config vars* akan menjadi `HEROKU_POSTGRESQL_<COLOR>_URL` dengan `<COLOR>` adalah nama warna yang dihasilkan secara acak oleh Heroku. Contoh: `HEROKU_POSTGRESQL_BLUE_URL`.

Apabila basis data yang digunakan lebih dari satu, basis data utama dapat diatur dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku pg:promote <database_url>
```

Keterangan:

- `<database_url>`: URL dari basis data.

Apabila pengguna Heroku ingin berbagi Heroku Postgres kepada banyak aplikasi, pengguna Heroku dapat mengetikkan perintah berikut pada *command shell* :

```
$ heroku addons:attach <originating_app>::DATABASE --app <receiver-app>
```

Keterangan:

- `<originating_app>`: nama aplikasi yang memiliki basis data yang ingin dibagi ke aplikasi lain.
- `<receiver-app>`: nama aplikasi yang akan menerima basis data dari aplikasi lain.

Pengguna Heroku dapat berhenti berbagi basis data dengan mengetikkan perintah berikut pada *command shell* :

```
$ heroku addons:detach <database_url> --app <application_name>
```

Keterangan:

- `<database_url>`: URL dari basis data
- `<application_name>`: nama aplikasinya.

Berikut adalah perintah-perintah dasar dari Heroku Postgres yang dapat diketikkan pada *command shell*:

- Melihat semua basis data milik aplikasi dan karakteristiknya:

```
$ heroku pg:info
```

- Mengawasi status basis data secara terus menerus:

```
$ watch heroku pg:info
```

- Mengadakan sesi `psql` dengan basis data:

```
$ heroku pg:psql
```

atau

```
$ heroku pg:psql <database_name>
```

Keterangan:

- <database_name> diisi dengan nama basis data atau cukup warna basis data, misalnya gray.
- Menarik data dari basis data Heroku Postgres ke basis data di mesin lokal:

```
$ heroku pg:pull
```

- Memasukkan data dari basis data di mesin lokal ke basis data di Heroku Postgres:

```
$ heroku pg:push <nama_db_lokal> <nama_db_heroku> --app <nama_aplikasi>
```

Keterangan:

- <nama_db_lokal>: nama basis data di lingkungan lokal.
- <nama_db_heroku>: nama basis data di lingkungan Heroku.

- Melihat daftar *query* yang berjalan:

```
$ heroku pg:ps
```

- Menghentikan *query* yang berjalan:

```
$ heroku pg:kill <procpid>
```

Keterangan:

- <procpid>: id *query*, didapat dari melihat daftar *query* yang berjalan.

- Menghentikan query yang berjalan secara paksa:

```
$ heroku pg:kill --force <procpid>
```

Keterangan:

- <procpid>: id *query*, didapat dari melihat daftar *query* yang berjalan.

- Menghentikan semua query yang berjalan:

```
$ heroku pg:killall
```

- Menghapus semua data di dalam basis data:

```
$ heroku pg:reset <database\_name>
```

Keterangan:

- <database_name> diisi dengan nama basis data atau cukup warna basis data, misalnya `gray`.

Basis data Heroku Postgres dapat diakses secara langsung menggunakan aplikasi pihak ketiga, misalnya PGAdmin. Informasi yang dibutuhkan untuk dapat terkoneksi basis data menggunakan aplikasi tersebut didapatkan dengan mengetikkan perintah :

```
$ heroku pg:credentials DATABASE
```

atau

```
$ heroku config | grep HEROKU_POSTGRESQL
```

Pada saat akan melakukan koneksi ke basis data, pengaturan SSL harus diatur ke `sslmode=require`.

2.1.5.2 Heroku Redis

Heroku Redis adalah basis data berbasis *key-value store* yang bersifat *in-memory*. Heroku Redis dapat diakses oleh bahasa apapun dengan *Redis driver*. Cara memasang *add-ons* Heroku Redis pada *command shell*:

```
$ heroku addons:create heroku-redis: <PLAN_NAME>
```

Keterangan :

- <PLAN_NAME>: nama *plan* Heroku Redis yang ingin dipakai. Heroku Redis memiliki dua pilihan *plan*: *Hobby Dev* dan *Premium*. *Hobby Dev* gratis, sedangkan *Premium* berbayar. Perbedaan kedua *plan* terletak pada jumlah memori dan batas koneksi yang dapat dibuat.

Heroku Redis memiliki kelebihan sebagai berikut:

- Memiliki analisa performa yang dapat membantu menemukan masalah basis data dengan mudah.
- Heroku Redis dapat diatur jumlah dan ukurannya sesuai kebutuhan memori dan koneksi.

2.1.5.3 Apache Kafka

Apache Kafka adalah salah satu *add-ons* di Heroku yang disediakan oleh Kafka yang berintegrasi penuh dengan Heroku. Apache Kafka dideskripsikan Heroku sebagai *add-ons* yang memungkinkan *developer* mendistribusikan aplikasi yang dapat menangani jutaan *event* dan miliaran transaksi. Kafka didesain untuk memindahkan data yang sangat besar dengan reliabilitas yang tinggi dan toleran akan kerusakan.

Pemasangan Python 2.7, node 8.x, .NET Framework, dan Visual C++ Build Tools diperlukan sebelum memasang Apache Kafka. Pemasangan Apache Kafka dapat dilakukan dengan mengetikkan perintah berikut pada *command shell*:

```
$ heroku plugins:install heroku-kafka
```

2.1.6 Verifikasi Akun

Untuk mengakses beberapa layanan di Heroku, pengguna Heroku harus melakukan verifikasi akun terlebih dahulu. Verifikasi akun dilakukan dengan memasukkan informasi kartu kredit atau kartu debit pengguna. Kartu kredit yang diterima oleh Heroku adalah kartu Visa, MasterCard, American Express, Discover, dan JCB. Kartu debit yang diterima adalah kartu Visa, MasterCard, dan JCB. Kartu lain tidak diterima. Beberapa bank mungkin mensyaratkan penahanan satu dollar oleh pelaku verifikasi sebelum kartu dapat dikonfirmasi.

Berikut beberapa layanan di Heroku yang membutuhkan verifikasi akun:

- Menggunakan lebih dari satu *dyno* di dalam aplikasi.
- Menambah *add-ons* selain Heroku Postgres dan Heroku Connect.
- Mengubah *domain* aplikasi.
- Menerima transfer dari aplikasi yang memiliki sumber daya berbayar.
- Menambah batas standar penggunaan *one-off dyno*.
- Memiliki lebih dari 5 aplikasi dalam satu waktu. Akun yang terverifikasi dapat memiliki sampai 100 aplikasi.

Cara melakukan verifikasi akun Heroku:

- Masuk ke halaman *Account Settings* (<https://dashboard.heroku.com/account>)
- Pilih menu *Billing*
- Tekan tombol *Add Credit Card*

2.2 Cron [2]

Cron adalah program yang menjalankan *command* (perintah) yang terteta di file *crontab* pada jadwal tertentu di latar belakang secara otomatis. File *crontab* dapat ditemukan di direktori */var/spool/cron/crontabs*. *Crontab* di direktori ini sebaiknya tidak diakses secara langsung, melainkan menggunakan *command crontab*. File *crontab* juga dapat ditemukan di direktori */etc* atau subdirektori */etc*. File *crontab* di direktori tersebut adalah *file system crontab*.

Setiap *command* di *crontab* diawali dengan lima penanda waktu dan diikuti dengan nama *user* (jika berada di *file system crontab*). Lima penanda waktu tersebut secara berurutan adalah:

- *minute*: menandakan tiap menit berapa *command* dijalankan. *Value* yang sah adalah angka 0-59 atau tanda *asterisk* (*).
- *hour*: menandakan tiap jam berapa *command* dijalankan. *Value* yang sah adalah angka 0-23 atau tanda *asterisk* (*).
- *day of the month*: menandakan tiap tanggal berapa *command* dijalankan. *Value* yang sah adalah angka 1-31 atau tanda *asterisk* (*).
- *month*: menandakan tiap bulan berapa *command* dijalankan. *Value* yang sah adalah angka 1-12 atau tanda *asterisk* (*).
- *day of the week*: menandakan tiap hari apa *command* dijalankan. *Value* yang sah adalah angka 0-7 atau tanda *asterisk* (*) atau "Sun". Hari minggu dapat ditulis dengan angka 0 atau 7 atau "Sun".

Contoh penulisan *command* di *crontab*:

```
0 12 * * * /home/myscripts/lunch.sh
```

Pada contoh di atas, *cron* akan menjalankan *script* *lunch.sh* pada direktori */home/myscripts* setiap hari pada pukul 12.00. Tanda asterisk menandakan eksekusi dilakukan dari *range value* terendah sampai *value* tertinggi.

Penanda waktu boleh memiliki *value* lebih dari satu. Penanda waktu yang memiliki *value* lebih dari satu ditulis dalam bentuk *range*, *list*, atau *step*. *Range* ditulis dalam format dua *value* yang dipisahkan dengan tanda hubung (-). Range bersifat inklusif. Contoh: 8-11 untuk *hour* berarti *command* akan dieksekusi pada pukul 8, 9, 10, dan 11. *List* ditulis dalam format dua atau lebih *value* yang dipisahkan dengan koma. Contoh: 1,5,7. *Range* dan *list* dapat disatukan. Contoh: 1,2,8-12. *Step* ditulis dalam format: *range* / <number>. Contoh : '0-23/2' untuk *hour*. Tanda tersebut berarti eksekusi dijalankan setiap dua jam sekali. Tanda asterisk juga dapat digunakan untuk menggantikan *range*. Contoh: */4 untuk *hour*. Tanda tersebut berarti eksekusi dijalankan setiap empat jam sekali.

Nama juga dapat digunakan untuk penanda waktu *month* atau *day of week*. Cukup dengan menggunakan tiga huruf pertama dari kata bahasa Inggrisnya tanpa menghiraukan huruf besar atau huruf kecil. Namun, nama tidak bisa dipakai untuk *range* atau *list*.

Selain itu, kelima penanda waktu dapat diganti dengan salah satu "nickname" berikut :

- @reboot: Eksekusi sekali saat *startup*.
- @yearly: Eksekusi setiap tahun (setiap tanggal 1 Januari), "0 0 1 1 *"
- @anually: Sama seperti @yearly.
- @monthly: Eksekusi setiap bulan, "0 0 1 * *".
- @weekly: Eksekusi setiap minggu, "0 0 * * 0".
- @daily: Eksekusi setiap hari (pukul 00.00), "0 0 * * *"
- @midnight: Sama seperti @daily.
- @hourly: Eksekusi setiap jam, "0 * * * *"

Baris kosong, spasi di awal baris, dan *tab* akan diabaikan oleh *cron*. Baris yang karakter bukan spasi pertamanya adalah # akan dianggap sebagai *comment* (komentar) dan akan diabaikan. *Comment* tidak boleh diletakkan di baris yang sama dengan *command* karena *comment* tersebut akan dianggap sebagai bagian dari *command*.

2.3 Gmail API [3]

Gmail adalah layanan *email* yang disediakan oleh perusahaan Google LLC. Gmail API dapat digunakan untuk mengakses *email* Gmail.

2.3.1 Resource

Gmail API menyediakan beberapa jenis *resource* :

- *Message*

Message merepresentasikan pesan dalam email. *Message* hanya bisa dibuat atau dihapus. Tidak ada properti dari *message* yang bisa diubah selain label yang diberikan ke message.

- *Label*

Label berfungsi sebagai sarana utama untuk mengelompokkan dan mengatur *message* dan *thread*. *Label* mempunyai hubungan banyak ke banyak dengan *message*. Artinya, satu *message* dapat memiliki beberapa *label* dan satu *label* dapat diberikan ke beberapa *message*.

Label dikelompokkan ke dalam dua kelompok: *label* sistem dan *label* pengguna. Contoh *label* sistem adalah *label INBOX*, *TRASH*, dan *SPAM*. *Label* sistem dibuat secara internal dan tidak dapat dibuat, dihapus, dan dimodifikasi. Namun, beberapa *label* sistem dapat diberikan ke *message* atau dilepaskan dari *message*. *Label* pengguna dapat ditambah, dihapus, dan dimodifikasi oleh pengguna atau aplikasi.

- *Draft*

Draft merepresentasikan *message* yang belum dikirim. *Message* tidak bisa dimodifikasi setelah dibuat, tapi *message* yang terdapat di dalam *draft* dapat dimodifikasi. Mengirimkan *draft* secara otomatis akan menghapus *draft* tersebut dan membuatnya menjadi *message* dengan *label* sistem *SENT*.

- *History*

History adalah riwayat modifikasi *message* yang diurutkan secara kronologis. *History* hanya menyimpan perubahan dalam jangka waktu 30 hari.

- *Thread*

Thread adalah kumpulan *message* yang merepresentasikan percakapan. *Thread* dapat memiliki *label*. *Thread* tidak dapat dibuat, tapi dapat dihapus. *Message* dapat dimasukkan ke *Thread*.

- *Setting*

Setting mengontrol perilaku fitur pada Gmail kepada penggunanya. *Setting* tersedia untuk akses POP dan IMAP, *forward email*, *filter*, *vacation auto-response*, *send-as aliases*, *signatures*, dan *delegates*.

2.3.2 Scope

Gmail API menggunakan OAuth 2.0 untuk menangani autentikasi dan *authorization*. Untuk menggunakan Gmail API, aplikasi harus menyebutkan *scope* yang dipakai di aplikasi. *Scope* adalah *string* yang mengidentifikasi *resource* yang ingin di akses. *Scope* ini digunakan bersama dengan token untuk mengamankan akses ke *resource* pengguna. *Token* tersebut memiliki masa kadaluarsa. Contoh *scope*:

- <https://www.googleapis.com/auth/gmail.readonly>: *scope* untuk membaca *message* dari Gmail.
- <https://www.googleapis.com/auth/gmail.modify>: *scope* untuk mengubah *label* pada *thread* atau *message*.
- <https://www.googleapis.com/auth/gmail.compose>: *scope* untuk mengirim *message* mewakili pengguna.

2.3.3 Penggunaan pada umumnya

Pada umumnya, GMail API digunakan untuk mengirim *message*, mengambil *email* yang diterima, melihat perubahan di *history*, dan mengatur *Label*.

2.3.3.1 Mengirim *message*

Berikut tahapan yang harus dilakukan untuk mengirim *message*:

1. Membuat konten *email*
2. Membuat *string* yang dikodekan berdasarkan *base64url* dari konten
3. Membuat *resource message* dan memasukkan *string* tersebut ke properti **raw**
4. Memanggil **message.send** untuk mengirim *message*

2.3.3.2 Mengambil *email* yang diterima

Pada Gmail API, mengambil *email* yang diterima dapat dilakukan dengan melakukan GET *request* ke "<https://www.googleapis.com/gmail/v1/users/userId/messages/id>" dengan **userId** adalah alamat *email* pengguna dan **id** adalah ID *email*. Saat mengambil message, format dari respon dapat diatur. Format FULL mengembalikan seluruh informasi dari message. Format MINIMAL hanya mengembalikan metadata seperti label. Format RAW mengembalikan properti **raw** saja. Secara otomatis, format dari respon memakai format FULL.

2.3.3.3 Perubahan di *history*

Perubahan *message* direpresentasikan oleh *History objects*. Properti **start_history_id** memperbolehkan aplikasi mengatur dari titik mana perubahan ingin dikembalikan. Beberapa perubahan dapat mempengaruhi lebih dari satu *message*, sehingga *history* yang merepresentasikan perubahan tersebut akan berisi beberapa *message*.

2.3.3.4 Manajemen *Label*

Label yang diberikan ke sebuah *thread* juga diberikan ke semua *message* di dalam *thread*. Jika sebuah *label* dihapus, *label* tersebut akan dihapus dari semua *thread* dan *message* yang memiliki *label* tersebut. Properti **messageListVisibility** digunakan untuk menentukan apakah *message* dengan *label* tersebut ada di *message list*. Properti **labelListVisibility** digunakan untuk menentukan apakah ada *label* tersebut di daftar *label*. Untuk mengubah *label*, gunakan **messages.modify** dan **threads.modify**.

2.3.4 Implementasi Otorisasi dari Sisi *Server*

Setiap *request* ke Gmail API harus menggunakan OAuth 2.0. *Developer* perlu menggunakan alur dari sisi *server* ketika aplikasinya membutuhkan akses Google API mewakili *user*. Pendekatan ini membutuhkan *access token* dan *refresh token* untuk *server*. Untuk mulai menggunakan Gmail API, *developer* harus mendapatkan *client id* dan *client secret* terlebih dahulu. *Client id* dan *client secret* dapat dimiliki apabila *developer* telah membuat *project* di Google API *Console* dan menyalakan Gmail API.

Ketika *user* membuka aplikasi untuk pertama kalinya, sebuah *dialog* akan muncul dan menanyakan izin dari *user* agar aplikasi boleh mengakses akun Gmail miliknya. *Dialog* tersebut akan menyatakan *scope* yang dipakai aplikasi. Setelah *user* mengizinkan, *dialog* tersebut tidak akan muncul lagi, kecuali *scope* aplikasi diubah.

Setelah *user sign-in* untuk pertama kalinya, *authorization result object* akan dikembalikan ke *server*. *Object* tersebut berisi *authorization code*. *Authorization code* adalah *code* sekali pakai yang dapat ditukar dengan *access token*. *Access token* ini akan diberikan ke Gmail API agar aplikasi diberi izin untuk mengakses ke data user pada waktu yang terbatas. Selain *access token*, *server* juga mendapatkan *refresh token*. *Refresh token* ini dapat digunakan untuk menerima *access token* baru setelah *token* yang lama kadaluarsa. *Refresh token* ini harus disimpan di suatu tempat agar

bisa dipakai. Jika *refresh token* ini tidak disimpan, maka aplikasi harus mengirim *request* dengan *query approval_prompt* yang diset ke *force*. Ini dapat mengakibatkan *user* mendapat dialog untuk meminta izin lagi.

Refresh token dapat kadaluarsa. *Refresh token* kadaluarsa jika :

- User mencabut izinnya.
- *Refresh token* sudah tidak digunakan selama enam bulan.
- User mengganti *password* dan *refresh token* berisi Gmail *scopes*.
- Akun *user* telah melebihi batas maksimal dari *refresh token* yang diizinkan. Batas maksimalnya adalah 50 *refresh token* per akun per klien. Jika batas ini dilampaui, membuat *refresh token* baru akan menggugurkan *refresh token* yang lama tanpa peringatan. Batas ini tidak berlaku untuk *service account*.

2.4 PHP IMAP [4]

IMAP (Internet Message Access Protocol) adalah metode untuk mengakses pesan elektronik yang disimpan di sebuah *mail server*.

Ekstensi ini dapat digunakan apabila *c-client library* sudah terpasang. Library ini dapat ditemukan di <https://www.washington.edu/imap/>. Dokumen IMAP tidak boleh diletakkan langsung ke dalam direktori *system*, karena dapat memicu konflik. Sebaiknya membuat direktori baru di dalam direktori *system*, lalu memasukkan *file* IMAP ke dalamnya. Contoh : /usr/local/imap-2000b. Di dalam direktori baru tambahkan direktori lagi bernama *lib/* dan *include/*. Semua *file* dengan ekstensi .c dimasukkan ke direktori *lib/*. Saat IMAP dikompilasi, *file* bernama *c-client.a* akan terbentuk. Dokumen tersebut juga diletakkan di direktori *lib/*.

Setelah itu, kompilasi PHP dengan *-with-imap[=DIR]* dengan DIR adalah tempat c-client. Contoh: *with-imap=/usr/local/imap-2000b*. Pengguna sistem operasi Windows mungkin harus mengaktifkan *php_imap.dll*.

IMAP tidak didukung pada sistem operasi Windows yang versinya lebih lama dari Windows 2000. Hal ini karena IMAP menggunakan fungsi enkripsi agar koneksi lewat SSL ke *mail server* aktif.

Di dalam sistem operasi Ubuntu, pemasangan PHP IMAP bisa dilakukan dengan mudah.

```
// Pasang libc-client-dev
$ sudo apt-get install libc-client-dev

// Pasang PHP<versi> imap:
// sudo apt-get install php<versi>-imap
// Contoh :
sudo apt-get install php5-imap
```

Berikut adalah *function* dasar dari imap:

- *imap_alerts*
 - Deskripsi: Fungsi ini mengembalikan semua *alert message* yang telah terjadi. Ketika fungsi ini dipanggil, semua *alert message* yang ada di *stack* dihapus.
 - *Parameter*: Tidak ada.
 - *Return values*: Mengembalikan *array* yang berisi semua *alert message* yang dihasilkan atau FALSE jika tidak ada satupun *alert message*.

- `imap_close`
 - Deskripsi: Fungsi ini berfungsi untuk menutup *IMAP stream*.
 - *Parameter*:
 - * `imap_stream`: *IMAP stream* yang dikembalikan oleh `imap_open`.
 - * `flag`: Jika diatur ke `CL_EXPUNGE`, fungsi akan secara diam-diam menghapus semua pesan yang ditandai untuk dihapus sebelum menutup *IMAP stream*.
 - *Return values*: Mengembalikan TRUE jika sukses atau FALSE jika gagal.
- `imap_errors`
 - Deskripsi: Fungsi ini mengembalikan semua *error* yang telah terjadi. Ketika fungsi ini dipanggil, semua *error* yang ada di *stack* dihapus.
 - *Parameter*: Tidak ada.
 - *Return values*: Mengembalikan *array* yang berisi semua *error* yang dihasilkan atau FALSE jika tidak ada satupun *error*.
- `imap_fetch_overview`
 - Deskripsi: Fungsi ini mengambil *mail header* berdasarkan urutan yang diberikan dan mengembalikan ikhtisar kontennya.
 - *Parameter* :
 - * `imap_stream`: *IMAP stream* yang dikembalikan oleh `imap_open`.
 - * `sequence`: Deskripsi cara pengurutan message. Cara penyebutan urutan bisa menggunakan sintaks X,Y atau mengambil semua dalam interval dengan sintaks X:Y.
 - * `options`: *sequence* akan berisi *message index* atau UID, jika parameter ini diatur ke `FT_UID`.
 - *Return values*: Mengembalikan *array of objects*. Tiap *object* mendeskripsikan satu *message header*. *Object* berisi macam-macam *property*. *Object* hanya akan menyebutkan sebuah *property* jika *property* tersebut memang ada. *Property* yang mungkin adalah :
 - * `subject`: Subjek pesan
 - * `from`: Pengirim
 - * `to`: Penerima
 - * `date`: Tanggal pengiriman
 - * `message_id`: *Message id*
 - * `references`: *Message id* yang behubungan
 - * `in_reply_to`: *Message id* untuk membalas
 - * `size`: ukuran dalam *bytes*
 - * `uid`: UID yang dimiliki di dalam *mailbox*
 - * `msgno`: urutan message di dalam *mailbox*
 - * `recent`: menandakan bahwa *message* ini adalah *message* yang baru-baru ini diterima
 - * `flagged`: menandakan bahwa *message* ini adalah *message* yang ditandai
 - * `answered`: menandakan bahwa *message* ini adalah *message* yang ditandai sebagai telah dijawab
 - * `deleted`: menandakan bahwa *message* ini adalah *message* yang ditandai untuk dihapus
 - * `seen`: menandakan bahwa *message* ini adalah *message* yang sudah dibaca
 - * `draft`: menandakan bahwa *message* ini adalah *message* yang ditandai sebagai *draft*

- * update: UNIX *timestamp* dari tanggal kedatangan pesan
- imap_fetchbody
 - Deskripsi: Mengambil bagian tertentu dari *body* dari *message* yang disebutkan. Bagian dari *body* tidak di-*decode* oleh fungsi ini.
 - Parameter:
 - * imap_stream: *IMAP stream* yang dikembalikan oleh imap_open.
 - * msg_number: nomor *message*
 - * section: Nomor bagian. Ini adalah serangkaian bilangan bulat yang dibatasi oleh periode yang diindeks ke daftar bagian *body* sesuai spesifikasi IMAP4.
 - * options: *bitmask* dengan satu atau lebih dari :
 - FT_UID: msg_number adalah UID
 - FT_PEEK: Jangan memberikan *seen flag* jika belum diberikan.
 - FT_INTERNAL: Mengembalikan *string* di dalam format internal, tidak akan dikanonikkan ke CRLF.
 - Return values: Mengembalikan bagian tertentu dari *body message* yang disebutkan sebagai *text string*.
- imap_fetchheader
 - Deskripsi: Fungsi ini mengambil *header* yang lengkap dan tidak terfilter (RFC2822 format) dari *message*.
 - Parameter :
 - * imap_stream: *IMAP stream* yang dikembalikan oleh imap_open.
 - * msg_number: nomor *message*
 - * options: pilihan yang mungkin adalah :
 - FT_UID: msgno argument adalah UID
 - FT_INTERNAL: Mengembalikan *string* di dalam format internal, tidak akan dikanonikkan ke CRLF.
 - FT_PREFETCHTEXT: RFC822.TEXT harus diambil sebelumnya pada saat yang sama. Ini menghindari RTT tambahan pada koneksi IMAP jika teks pesan lengkap diinginkan.
 - Return values: Mengembalikan header dari message yang disebutkan sebagai text string.
- imap_fetchstructure
 - Deskripsi: Mengambil semua informasi terstruktur untuk message yang diberikan.
 - Parameter :
 - * imap_stream: *IMAP stream* yang dikembalikan oleh imap_open.
 - * msg_number: nomor *message*
 - * options: parameter opsional ini hanya memiliki satu opsi, FT_UID, yang memberitahu fungsi untuk memperlakukan msg_number argument sebagai UID.
 - Return values: Mengembalikan sebuah *object* termasuk *envelope*, *internal date*, *size*, *flags* dan *body structure* serta *object* serupa untuk tiap *mime attachment*. Struktur dari *object* adalah sebagai berikut :
 - * type: *Primary body type*
 - * encoding: *Body transfer encoding*
 - * ifsubtype: TRUE jika ada *subtype string*

- * subtype: *MIME subtype*
- * ifdescription: TRUE jika ada *description string*
- * description: *Content description string*
- * ifid: TRUE jika ada *identification string*
- * id: *Identification string*
- * lines: Jumlah *lines*
- * bytes: Jumlah *bytes*
- * ifdisposition: TRUE jika ada *disposition string*
- * disposition: *Disposition string*
- * ifdparameters: TRUE jika *dparameters array* tersedia
- * dparameters: *Array of objects* dimana tiap *object* memiliki "attribute" dan "value" *property* berdasarkan parameter pada *Content-disposition MIME header*.
- * ifparameters: TRUE jika *parameters array* tersedia
- * parameters: *Array of objects* dimana tiap *object* memiliki "attribute" dan "value" *property*.
- * parts: *Array of objects* identik dalam *structure* dengan *top-level object*, masing-masing berdasarkan pada *MIME body part*.

- **imap_headerinfo**

- Deskripsi: Fungsi ini berfungsi untuk mendapatkan informasi dari *message number* yang diberikan dengan membaca *header*.
- *Parameter* :
 - * imap_stream: *IMAP stream* yang dikembalikan oleh `imap_open`.
 - * msg_number: nomor *message*
 - * fromlength: jumlah karakter untuk *fetchfrom property*. Harus lebih besar atau sama dengan nol.
 - * subjectlength: jumlah karakter untuk *fetchsubject property*. Harus lebih besar atau sama dengan nol.
 - * defaulthost
- *Return values*: Mengembalikan FALSE jika terjadi *error*. Jika sukses, mengembalikan informasi di dalam *object* dengan *property* berikut :
 - * toaddress: *full "to" : line*, sampai dengan 1024 karakter.
 - * to: *array of objects* dari *To: line*, dengan *property* berikut: *personal, adl, mailbox, dan host*
 - * fromaddress: *full "from" : line*, sampai dengan 1024 karakter.
 - * from: *array of objects* dari *From: line*, dengan *property* berikut: *personal, adl, mailbox, dan host*.
 - * ccaddress: *full "cc" : line*, sampai dengan 1024 karakter.
 - * cc: *array of objects* dari *Cc: line*, dengan *property* berikut: *personal, adl, mailbox, dan host*.
 - * bccaddress: *full "bcc" : line*, sampai dengan 1024 karakter.
 - * bcc: *array of objects* dari *Bcc: line*, dengan *property* berikut: *personal, adl, mailbox, dan host*.
 - * reply_toaddress: *full "Reply-To" : line*, sampai dengan 1024 karakter.
 - * reply_to: *array of objects* dari *Reply-To: line*, dengan *property* berikut: *personal, adl, mailbox, dan host*.
 - * senderaddress: *full "sender" : line*, sampai dengan 1024 karakter.

- * sender: *array of objects* dari *Sender: line*, dengan *property* berikut: *personal, adl, mailbox, dan host.*
 - * return_pathaddress: *full "Return-Path": line*, sampai dengan 1024 karakter.
 - * return_path: *array of objects* dari *Return-Path: line*, dengan *property* berikut: *personal, adl, mailbox, dan host.*
 - * remail
 - * date: tanggal *message* yang ditemukan di *header*
 - * Date: sama dengan *date*
 - * subject: subyek pesan
 - * Subject: sama dengan *subject*
 - * in_reply_to
 - * message_id
 - * newsgroups
 - * followup_to
 - * references
 - * Recent: R jika diterima baru-baru ini dan telah dilihat, N jika diterima baru-baru ini dan belum pernah dilihat, dan '' jika tidak diterima baru-baru ini.
 - * Unseen: U jika belum pernah dilihat dan tidak diterima baru-baru ini, '' jika telah dilihat atau belum pernah dilihat dan diterima baru-baru ini.
 - * Flagged: F jika telah ditandai, '' jika tidak ditandai.
 - * Answered: A jika telah dijawab, '' jika belum dijawab.
 - * Deleted: D jika telah dihapus, '' jika belum dihapus.
 - * Draft: X jika *message* adalah sebuah *draft*, '' jika bukan *draft*.
 - * Msgno: nomor *message*
 - * MailDate
 - * Size: ukuran *message*
 - * udate: tanggal *message* diterima dalam format waktu Unix.
 - * fetchfrom: *from line* diformat untuk memenuhi *fromlength characters*.
 - * fetchsubject: *subject line* diformat untuk memenuhi *subjectlength characters*.
- imap_last_error
 - Deskripsi: Fungsi ini berfungsi untuk mengembalikan *full text* dari IMAP *error message* terakhir yang terjadi pada *page* sekarang. *Error stack* tidak diganggu-gugat.
 - Parameter: Tidak ada
 - Return values: Mengembalikan *full text* dari IMAP *error message* terakhir yang terjadi pada *page* sekarang. Mengembalikan FALSE jika tidak ada *error message*.
 - imap_open
 - Deskripsi: Fungsi ini berfungsi untuk membuka *IMAP stream* ke sebuah *mailbox*. Fungsi ini dapat juga digunakan untuk membuka *stream* ke POP3 dan NNTP *server*, tapi beberapa fungsi dan fitur hanya tersedia pada *IMAP server*.
 - Parameter :
 - * mailbox: nama *mailbox* yang terdiri dari *server* dan *mailbox path* pada server ini.
 - * username
 - * password
 - * options: *bit mask* dengan satu atau lebih dari berikut:

- OP_READONLY: Membuka *mailbox*, *read-only*
 - OP_ANONYMOUS: Tidak menggunakan atau memperbarui `.newsrc` untuk *news* (hanya NNTP)
 - OP_HALFOPEN: Untuk nama IMAP dan NNTP, membuka koneksi tapi tidak membuka *mailbox*
 - CL_EXPUNGE: Menghapus pesan yang ditandai untuk dihapus sebelum menutup *mailbox*
 - OP_DEBUG: *Debug protocol negotiations*
 - OP_SHORTCACHE: *Short (elt-only) caching*
 - OP_SILENT: Jangan mengoper *events* (penggunaan internal)
 - OP_PROTOTYPE: Mengembalikan *driver prototype*
 - OP_SECURE: Jangan melakukan autentikasi yang tidak aman
- * n_retries: Jumlah maksimum percobaan untuk terkoneksi
 - * params: parameter koneksi, *string/key* berikut mungkin dapat digunakan untuk mengatur satu atau lebih parameter koneksi: DISABLE_AUTHENTICATOR (Menonaktifkan *property* autentikasi).
- *Return values*: Mengembalikan *IMAP stream* jika berhasil dan FALSE jika terjadi *error*.
- imap_qprint
 - Deskripsi: Menkonversi *quoted-printable string* ke dalam *8 bit string* berdasarkan RFC2045, section 6.7.
 - Parameter: *quoted-printable string*
 - *Return values*: *8 bit string*
 - imap_search
 - Deskripsi: Fungsi ini berfungsi untuk melakukan pencarian pada *mailbox* yang sedang terbuka pada *IMAP stream* yang diberikan.
 - Parameter:
 - * imap_stream: *IMAP stream* yang dikembalikan oleh `imap_open`.
 - * criteria: *string* yang dibatasi dengan spasi, di dalamnya berisi satu atau lebih kata kunci berikut:
 - ALL: mengembalikan semua *message* yang sesuai dengan kriteria lainnya.
 - ANSWERED: mengembalikan *message* yang bertanda ANSWERED
 - BCC "string": mengembalikan *message* dengan "string" di dalam *Bcc: field*
 - BEFORE "date": mengembalikan *message* dengan *Date*: sebelum "date"
 - BODY "string": mengembalikan *message* dengan "string" di dalam *body message*
 - CC "string": mengembalikan *message* dengan "string" di dalam *Cc: field*
 - DELETED: mengembalikan *message* yang dihapus
 - FLAGGED: mengembalikan *message* yang bertanda FLAGGED (terkadang merujuk ke *message* bertanda *Important* atau *Urgent*)
 - FROM "string": mengembalikan *message* dengan "string" di dalam *From: field*
 - KEYWORD "string": mengembalikan *message* dengan "string" sebagai *keyword*
 - NEW: mengembalikan *message* baru
 - OLD: mengembalikan *message* lama
 - ON "date": mengembalikan *message* dengan *Date*: cocok dengan "date"
 - RECENT: mengembalikan *message* yang baru-baru ini diterima atau yang bertanda RECENT

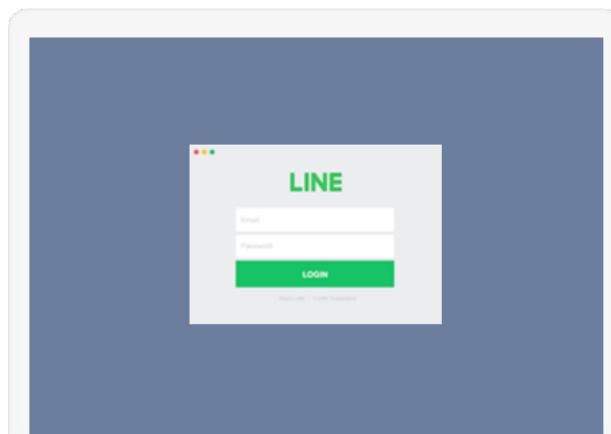
- SEEN: mengembalikan *message* yang telah dilihat atau yang bertanda SEEN
- SINCE "date": mengembalikan *message* dengan *Date*: sejak "date"
- SUBJECT "string": mengembalikan *message* dengan "string" di dalam *Subject*
- TEXT "string": mengembalikan *message* dengan text "string"
- TO "string": mengembalikan *message* dengan "string" di dalam *To*:
- UNANSWERED: mengembalikan *message* yang belum dijawab
- UNDELETED: mengembalikan *message* yang belum dihapus
- UNFLAGGED: mengembalikan *message* yang belum ada tandanya
- UNKEYWORD "string": mengembalikan *message* yang tidak memiliki *keyword* "string"
- UNSEEN: mengembalikan *message* yang belum pernah dilihat
- * options: *Values* sah untuk *options* adalah SE_UID, yang menyebabkan *array* yang dikembalikan berisi UID, bukan *message sequence number*.
- * charset: *MIME character set* untuk digunakan saat mencari *strings*.
- *Return values*: Mengembalikan *array* dari *message number* atau UID.

2.5 LINE [5]

LINE adalah aplikasi pengirim pesan yang tersedia dalam *platform android, ios, dan desktop*. LINE memiliki beberapa produk yang dapat digunakan *developer* aplikasi. Produk-produk tersebut adalah:

1. LINE Login
2. LINE Bot Designer
3. Clova
4. LINE Pay
5. Messaging API

2.5.1 LINE Login



Gambar 2.8: LINE Login

LINE Login adalah produk dari LINE yang memungkinkan *developer* membuat aplikasinya menyediakan pilihan *login* melalui akun LINE. Pengguna aplikasi yang dibuat *developer* tidak perlu mendaftar menggunakan *email* dan *password*. *Login* menjadi lebih mudah dan cepat. LINE menyediakan LINE SDK untuk mengintegrasikan LINE Login dengan *native apps*.

2.5.2 LINE Bot Designer



Gambar 2.9: LINE Bot Designer

LINE Bot Designer (Gambar 2.9) adalah produk LINE yang memungkinkan *developer* membuat prototipe LINE bot lebih cepat dan lebih mudah tanpa mengetahui pemrograman. Dengan produk ini, *developer* dapat mendesain *chatbots* sesuai skenario yang diinginkan.

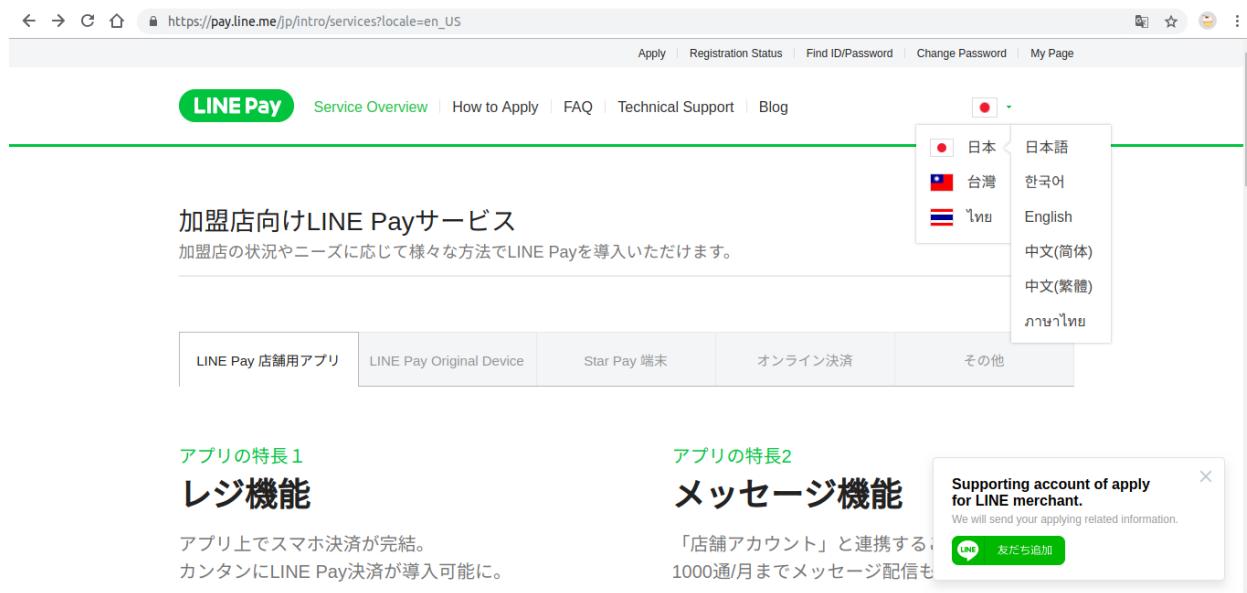
2.5.3 Clova



Gambar 2.10: Situs web Clova (<https://clova-developers.line.me>)

Clova adalah sebuah *AI Assistant* (aplikasi dengan kecerdasan buatan yang berfungsi sebagai asisten) yang dipasang di dalam *Clova Wave* dan *Clova Friends*. Pada saat skripsi ini dibuat, Clova masih dalam tahap pengembangan dan tersedia dalam versi beta. Tidak ada dokumentasi resmi untuk produk ini, tapi ada situs web resminya: <https://clova-developers.line.me> (Gambar 2.10). Pada saat skripsi ini ditulis, situs web ini hanya tersedia dalam bahasa Jepang sehingga membutuhkan penerjemah untuk memahami isinya.

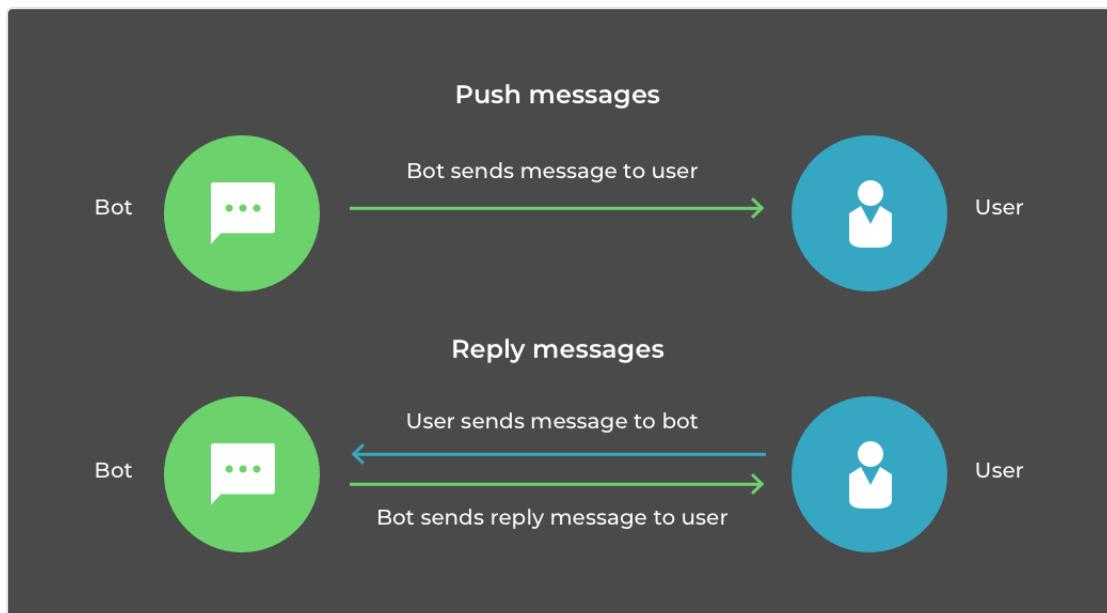
2.5.4 LINE Pay



Gambar 2.11: Situs web LINE Pay (<https://pay.line.me>)

LINE Pay adalah produk LINE yang memungkinkan *developer* mengintegrasikan aplikasi yang dibuat *developer* dengan fitur pembayaran melalui LINE Pay. Tidak ada dokumentasi resmi untuk produk ini, tapi ada situs web resminya: <https://pay.line.me> (Gambar 2.11). Situs web ini menyediakan informasi LINE Pay di negara Jepang, Republik Tiongkok / Taiwan, dan Thailand. Situs ini tersedia dalam bahasa Jepang, Korea, Inggris, China dengan aksara sederhana, China dengan aksara tradisional, dan Thailand.

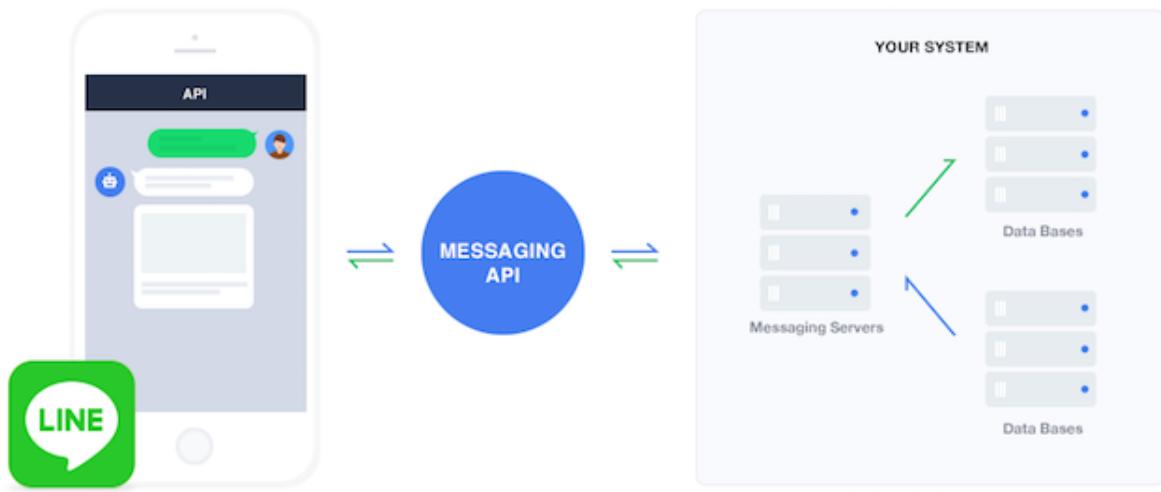
2.5.5 Messaging API



Gambar 2.12: Messaging API memungkinkan *developer* mengirim push message dan reply message

Messaging API adalah produk LINE yang memungkinkan *developer* untuk membangun *bot* sebagai sarana komunikasi dua arah antara layanan yang dibangun *developer* dengan pengguna LINE. Dengan Messaging API, *developer* dapat mengirimkan *push message* dan *reply message* (Gambar 2.12) ke akun LINE@. *Push message* adalah pesan yang *bot* kirimkan ke pengguna LINE. *Reply message* adalah pesan yang *bot* kirimkan untuk membalas pesan dari pengguna LINE.

LINE Menyediakan Messaging API untuk membangun messaging bot. Messaging API memungkinkan data dioper antara server dari aplikasi bot dengan LINE Platform. Ketika pengguna LINE mengirimkan pesan ke bot, sebuah *webhook* akan terpicu dan LINE Platform akan mengirimkan permintaan ke URL *webhook bot*. Server akan mengirim permintaan ke LINE Platform untuk merespon pengguna. Permintaan akan dikirimkan dalam format JSON. Arsitektur dari Messaging API dapat dilihat pada Gambar 2.13.



Gambar 2.13: Arsitektur Messaging API

Developer dapat melakukan hal-hal berikut dengan Messaging API :

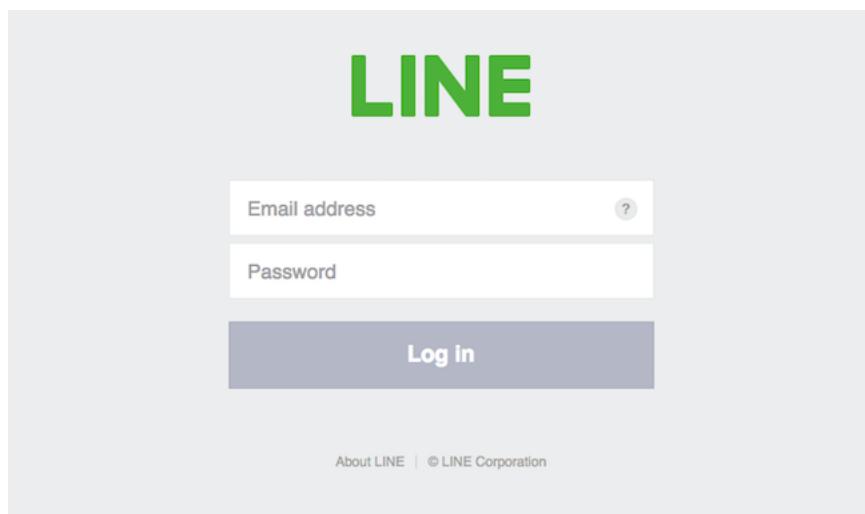
- Mengirimkan *reply message*
- Mengirimkan *push message*
- Mengirimkan berbagai jenis pesan
- Mendapatkan profil pengguna yang berinteraksi dengan *bot*
- Bergabung dengan percakapan grup (*group chats*)

Untuk menggunakan Messaging API, *developer* memerlukan akun LINE@. Messaging API juga dapat digunakan menggunakan akun resmi/*official accounts*. Akun resmi mendapatkan fitur tambahan untuk pengguna *enterprise*.

2.5.5.1 Membuat *Channel*

Untuk memulai membangun bot dengan Messaging API, *developer* perlu membuat *channel* terlebih dahulu. *Channel* adalah penyambung antara LINE platform dan aplikasi yang dibuat *developer*. Berikut langkah-langkah untuk membuat *channel*:

1. Langkah ke-1 : Masuk ke LINE Developers *console*



Gambar 2.14: Tampilan LINE Developers *console* saat login

Developer perlu masuk ke LINE Developers *console* (<https://developers.line.me/en/>) dengan alamat *email* dan *password* dari akun LINE *developer* (Gambar 2.14). Jika *developer* belum memiliki akun LINE, *developer* perlu mengunduh aplikasi LINE untuk mendaftar akun LINE.

2. Langkah ke-2 : Mendaftar sebagai *developer*

Gambar 2.15: Tampilan LINE Developers *console* saat *register developer*

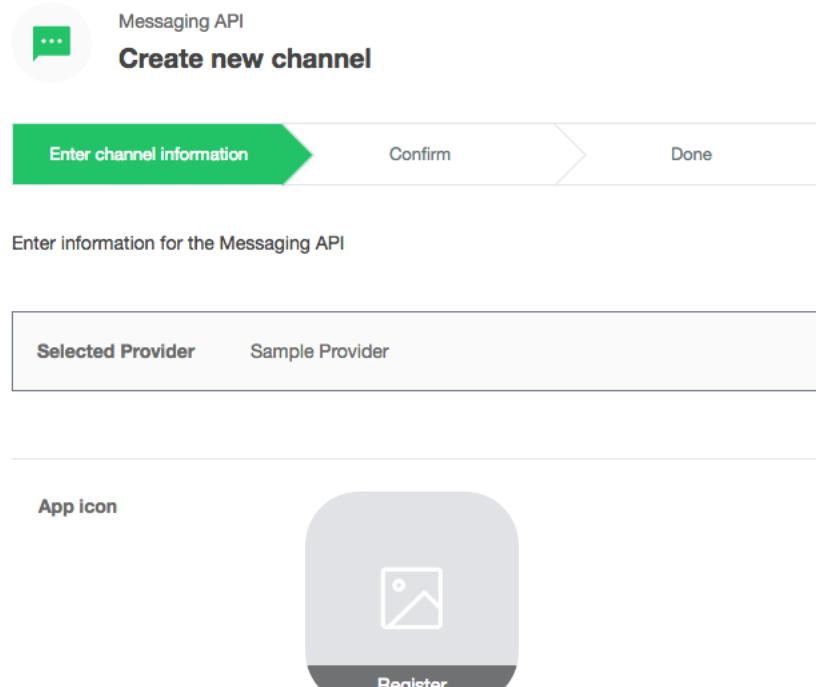
Apabila *developer* baru pertama kali masuk ke LINE Developers *console*, *developer* perlu membuat akun *developer* (Gambar 2.15). *Developer* hanya perlu mencantumkan nama dan

alamat *email* untuk mendaftar.

3. Langkah ke-3 : Membuat *provider* baru

Provider adalah individu atau perusahaan yang menyediakan aplikasi yang akan dibuat. *Developer* perlu mencantumkan nama *provider* untuk membuat *provider* baru. *Developer* dapat menuliskan nama *developer* sendiri atau nama perusahaan *developer*.

4. Langkah ke-4 : Membuat *channel*



Gambar 2.16: Tampilan LINE Developers *console* saat membuat *channel*

Developer perlu memasukkan informasi yang dibutuhkan untuk membuat *channel*:

- Ikon aplikasi
Dokumen gambar untuk ikon aplikasi harus dibawah 3 MB dengan ekstensi JPEG/PNG/GIF/BMP.
- Nama aplikasi
Nama aplikasi tidak boleh lebih dari 20 karakter. Kata "LINE" tidak dapat digunakan sebagai nama aplikasi, walaupun kapitalisasinya tidak sama. Setelah dikonfirmasi, nama aplikasi tidak dapat diubah untuk tujuh hari ke depan.
- Deskripsi aplikasi
Deskripsi aplikasi tidak boleh lebih dari 500 karakter.
- *Plan*
Pada saat mendaftar *channel*, pilihan *plan* yang tersedia hanya dua: *Developer Trial* dan *Free*. *Plan Developer Trial* memungkinkan *developer* untuk membuat bot yang dapat mengirimkan *push message* dan memiliki 50 teman. Apabila *developer* memilih plan ini, maka *developer* tidak dapat melakukan *upgrade* atau membeli ID *premium*. *Plan Free* memungkinkan *developer* untuk membuat bot dengan jumlah teman tak terbatas, tapi *developer* tidak dapat mengirimkan *push message*. *Developer* dapat melakukan *upgrade* kapan saja dengan *plan* ini.

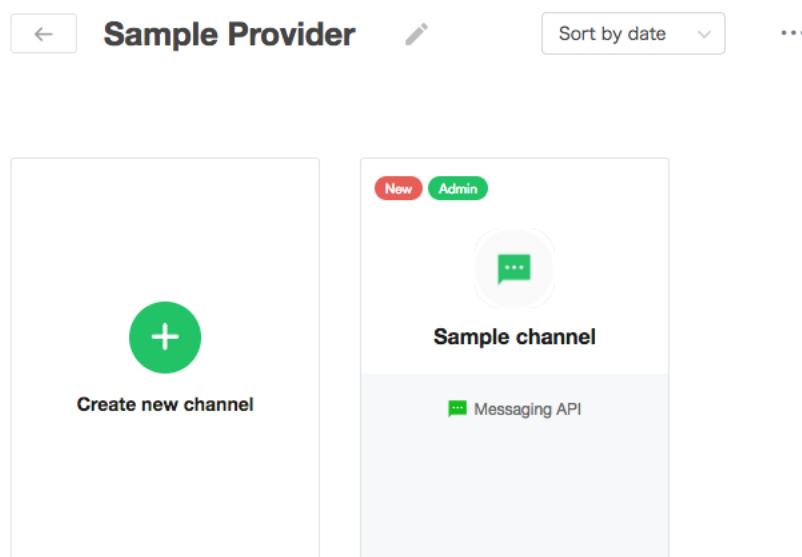
- Kategori dan Subkategori

Developer dapat memilih kategori dan subkategori yang cocok dengan aplikasi yang sedang dikembangkan.

- Alamat *email*

Alamat *email* yang dicantumkan adalah alamat *email* yang akan menerima notifikasi dan pengumuman penting dari LINE. Maksimal karakter pada alamat *email* adalah 100 karakter.

5. Konfirmasi



Gambar 2.17: Tampilan LINE Developers *console* saat konfirmasi pembuatan *channel*

Konfirmasi *channel* yang baru saja dibuat. Tampilan setelah *channel* dikonfirmasi dapat dilihat di Gambar 2.17.

2.5.5.2 Membuat *bot*

Setelah membangun *channel*, *developer* perlu menyiapkan *server* untuk menjadi *host* dari bot. *Developer* dapat menggunakan layanan *cloud platform*, seperti Heroku. Setelah itu, *developer* dapat mulai mengatur *bot* pada *console*.

Aplikasi *bot* membutuhkan *channel access token* untuk membuat *API call* dan *webhook URL* untuk menerima *webhook payload* dari LINE Platform. *Channel access token* adalah *long-lived token* (token yang tidak memiliki kadaluarsa) yang harus diatur di dalam *authorization header* ketika membuat *API call*. *Developer* dapat menerbitkan lagi *channel access token* kapanpun melalui *console*. Untuk menerbitkan *channel access token*, klik Issue pada "Channel settings" di halaman *console*. Sedangkan *webhook URL* adalah titik akhir dari *server* aplikasi *bot* dimana *webhook payload* dikirimkan.

Untuk mengatur *webhook URL*, *developer* dapat memasukkannya ke halaman *Channel settings* pada *console*. *Webhooks* harus diaktifkan terlebih dahulu dengan menekan tombol *enable webhooks*. Untuk memeriksa apakah *webhook URL* dapat menerima *event webhook*, tekan tombol *Verify* dan pastikan hasilnya "*Success*". *Webhook URL* harus menggunakan HTTPS dan memiliki sertifikat SSL yang diterbitkan oleh *certificate authority* (CA) yang terotorisasi.

Setelah *token* dan *webhook URL* berhasil diset, tambahkan *bot* sebagai teman melalui akun LINE. *Developer* dapat melakukannya dengan *scan* kode QR pada *Channel Settings*.

2.5.5.3 Menkonfigurasi Keamanan

Developer dapat mengkonfigurasi keamanan tapi tidak wajib dilakukan. Untuk meningkatkan keamanan, *developer* dapat mengatur *server* yang dapat memanggil API pada LINE Platform pada *Security settings*. *Developer* dapat mendaftarkan alamat IP secara individual atau jika *developer* memiliki *server* yang banyak *developer* dapat menggunakan notasi *Classless Inter-Domain Routing* (CIDR) untuk mendaftarkan alamat jaringan.

2.5.5.4 Messaging API SDK

LINE menyediakan *library*, *tool*, dan *sample* untuk mempermudah pembangunan aplikasi bot dengan Messaging API. Ketiga benda tersebut tergabung dalam SDK (*Software Development Kit*). LINE menyediakan SDK untuk bahasa Java, PHP, Go, Perl, Ruby, Python, dan Node.js.

2.5.5.5 Alur kerja Messaging API

Ketika *user* berinteraksi dengan *bot* seperti mengirimkan pesan atau menambah *bot* sebagai teman, LINE Platform mengirimkan HTTP POST *request* yang berisi *webhook event object* ke *bot server* yang disebutkan di kolom "*Webhook URL*" pada *console*. *Request header* berisi *signature*.

Untuk mengecek apakah *server* dapat menerima *webhook event*, blokir *bot* pada LINE dan cek *server logs* untuk menkonfirmasi bahwa *server* dapat menerima *unfollow event* dari LINE Platform.

Untuk memastikan *request* yang dikirim berasal dari LINE Platform, *bot server* harus memvalidasi *X-Line-Signature* pada *request header*. Caranya dengan :

1. Menggunakan *channel secret* sebagai *secret key*, menghasilkan *Base64-encoded digest* dari *request body* menggunakan algoritma HMAC-SHA256
2. Mengonfirmasi *signature X-Line-Signature* dalam *request header* cocok dengan *digest*.

2.5.5.6 Webhook Event Object

Webhook Event Object adalah objek JSON yang berisi event yang dihasilkan oleh *platform* LINE. *Webhook Event Object* yang dihasilkan tergantung dari jenis *chat*. Jenis *chat* ada dua, yaitu: *one-on-one chat* (*chat* antara dua akun LINE) dan *group chats* (*chat* di dalam grup). Berikut *webhook event object* yang muncul saat:

1. *one-on-one chat*:
 - *Message Event*
Menunjukkan bahwa ada *user* yang mengirim pesan. *Event* ini dapat dibalas.
 - *Follow Event*
Menunjukkan bahwa akun *bot* ditambahkan sebagai teman (atau dibuka blokirnya). *Event* ini dapat dibalas.
 - *Unfollow Event*
Menunjukkan bahwa akun *bot* diblokir
 - *Postback event*
Menunjukkan *user* melakukan aksi *postback*. *Event* ini dapat dibalas.
 - *Beacon event*
Menunjukkan bahwa *user* telah masuk atau keluar dari jangkauan LINE Beacon. *Event* ini dapat dibalas.

- *Account link event*

Menunjukkan bahwa *user* telah menghubungkan akun LINE dengan akun layanan *developer*.

2. *group chats*:

- *Message event*

Menunjukkan bahwa ada *user* yang mengirim pesan. *Event* ini dapat dibalas.

- *Join event*

Menunjukkan *bot* telah bergabung ke sebuah *group chat*

- *Leave event*

Menunjukkan *bot* telah keluar dari sebuah *group chat*

- *Postback event*

Menunjukkan *user* melakukan aksi *postback*. *Event* ini dapat dibalas.

2.5.5.7 Operasi pada *bot*

Developer dapat melakukan operasi berikut lewat *bot* :

1. Mengirim *reply message*

Reply message adalah pesan yang dikirim sebagai respons dari *user-generated event*. *User-generated event* adalah *event* yang muncul karena *user* berinteraksi dengan *bot*, misalnya mengirim pesan. *Developer* hanya dapat membalas *webhook events* yang memiliki *reply token*. Untuk membalas pesan, kirim HTTP POST *request* ke `/bot/message/reply`. Sertakan *channel access token* di dalam *authorization header* dan *reply token* di *request body*. *Developer* dapat mengirimkan sampai 5 *message object* per *request*.

2. Mengirim *push message*

Untuk mengirim *push message*, *developer* harus memerhatikan *plan* yang dipakai. Apabila *developer* memakai *plan Free* maka *developer* tidak dapat melakukan operasi ini. *Push message* adalah pesan yang dapat *bot* kirimkan ke *user* kapan saja. *Push message* tidak membutuhkan *reply token* seperti saat mengirim *reply message*. Ketika mengirim *push message*, sebutkan *user ID* di dalam *property to*. ID penerima dapat ditemukan dari *webhook event object*. Apabila penerima hanya satu, kirimkan *request* ke `/bot/message/push`. Sedangkan apabila penerima ada beberapa, kirimkan ke `/bot/message/multicast`. *Developer* dapat mengirimkan sampai 5 *message object* per *request*.

3. Mendapatkan konten yang dikirim oleh *user*

Untuk mengambil gambar, video, atau audio yang dikirim *user*, kirimkan HTTP GET *request* ke `/bot/message/messageId/content`. Konten yang dikirim oleh *user* otomatis dihapus dalam jangka waktu tertentu.

4. Mendapatkan informasi *user profile*

Untuk mendapatkan informasi *user profile* dari *user* yang menambahkan *bot* atau mengirim pesan ke *bot*, kirimkan HTTP GET *request* ke `/bot/profile/userId`. *Request* ini akan mengembalikan *display name*, *user ID*, *profile image URL*, dan *status message* (jika tersedia) dari *user*.

2.5.6 LINE@ Manager

LINE@ Manager adalah alat untuk mengatur akun LINE@ (LINE bot). *Developer* dapat meningkatkan *user experience* dengan mengatur halaman akun, membuat *Timeline post*, dan menggunakan fitur lain yang disediakan LINE@ Manager. Berikut adalah hal-hal yang bisa dilakukan:

1. Mengubah tampilan halaman akun

Developer dapat mengubah gambar cover, logo, tombol, dan informasi yang disediakan

2. Mengatur *greeting message*

Jika *developer* mengaktifkan *greeting message* pada *Channel settings*, maka *developer* dapat mengatur *greeting message* yang akan dikirim ke *user* saat pertama kali menambahkan bot sebagai teman. *Developer* dapat melakukannya juga dengan program melalui *follow webhook event*.

3. Mengatur *auto reply message*

Jika *developer* mengaktifkan "*Auto reply message*" pada *Channel settings*, maka *developer* dapat mengatur pesan balasan otomatis setiap *user* mengirimkan pesan ke bot.

2.6 System Usability Scale [6]

System Usability Scale adalah salah satu metode untuk menguji usabilitas dari sebuah sistem atau produk. Metode ini menggunakan kuesioner yang terdiri dari 10 pernyataan berikut:

1. *I think that I would like to use this system frequently.*
2. *I found the system unnecessarily complex.*
3. *I thought the system was easy to use.*
4. *I think that I would need the support of a technical person to be able to use this system.*
5. *I found the various functions in this system were well integrated.*
6. *I thought there was too much inconsistency in this system.*
7. *I would imagine that most people would learn to use this system very quickly.*
8. *I found the system very cumbersome to use.*
9. *I felt very confident using the system.*
10. *I needed to learn a lot of things before I could get going with this system.*

Setiap pernyataan dinilai oleh responden dengan angka di antara 1 sampai 5. Angka 1 menunjukkan responden tidak setuju dengan pernyataan tersebut dan angka 5 menunjukkan bahwa responden sangat setuju dengan pernyataan tersebut. Setelah seluruh responden mengisi kuesioner tersebut, skor *System Usability Scale* dari tiap responden dihitung. Cara menghitungnya adalah sebagai berikut:

1. Kurangi nilai pertanyaan bermotor ganjil dengan angka 1.
2. Kurangi 5 dengan nilai pertanyaan bermotor genap.
3. Jumlahkan semua hasilnya lalu kalikan dengan 2.5.

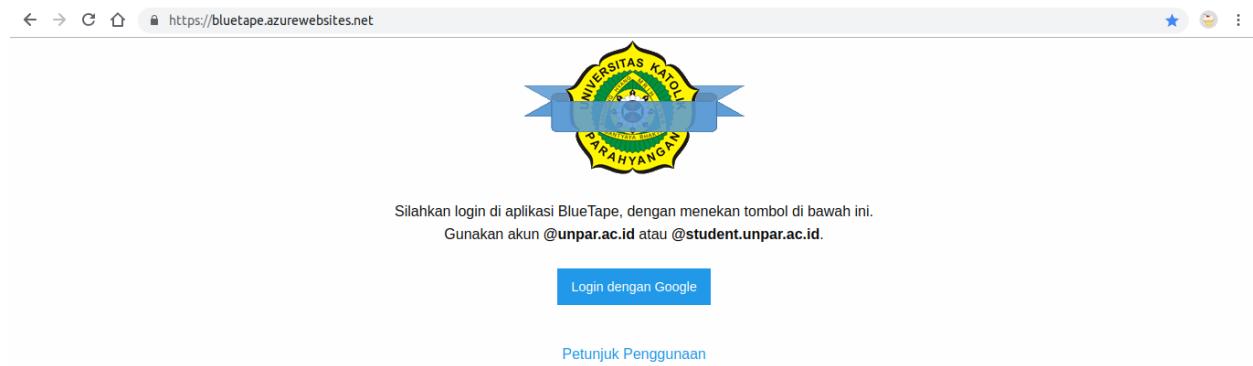
Menurut analisis Tullis(2008) dan Bangor, Kortum, dan Miller(2009), apabila rata-rata skor *System Usability Scale* kurang dari 50, maka usabilitas sistem atau produk termasuk jelek. Apabila rata-rata skor *System Usability Scale* di antara 50-70, maka usabilitas sistem atau produk termasuk kelompok rata-rata. Apabila rata-rata skor *System Usability Scale* lebih dari 70, maka usabilitas sistem atau produk termasuk bagus.

BAB 3

ANALISIS

Pengumpulan data dalam skripsi ini dilakukan dengan cara studi pustaka.

3.1 Analisis Sistem Kini



Gambar 3.1: Tampilan utama BlueTape

BlueTape adalah aplikasi yang berfungsi untuk membantu urusan-urusan *paper-based* di FTIS UNPAR menjadi *paperless*. Aplikasi ini berbasis web dengan memanfaatkan *framework* CodeIgniter (versi 3.1.4) dan ZURB Foundation. Selain itu, BlueTape menggunakan layanan OAuth dari Google untuk autentikasi pengguna. Saat skripsi ini ditulis, aplikasi BlueTape memiliki tiga layanan, yaitu Transkrip *Request / Manage*, Perubahan Kuliah *Request / Manage*, dan perekam jadwal dosen. Layanan Transkrip *Request / Manage* memberikan layanan untuk melakukan permohonan serta pencetakan transkrip mahasiswa. Layanan Perubahan Kuliah *Request / Manage* memberikan layanan untuk permohonan dan pencetakan perubahan jadwal kuliah oleh dosen. Layanan perekam jadwal dosen memberikan layanan untuk menyimpan dan menampilkan jadwal dosen.¹

Pada saat skripsi ini dibuat, BlueTape dapat diakses melalui situs web <https://bluetape.azurewebsites.net/> (Gambar 3.1). Perangkat lunak ini bersifat *open source*, sehingga kode program BlueTape bisa dipelajari, diubah, dan distribusi oleh siapapun untuk tujuan apapun. Kode program ini dapat diakses di <https://github.com/ftisunpar/BlueTape>.

Pola pengembangan yang dipakai BlueTape adalah MVC (*Model-View-Controller*). MVC (*Model-View-Controller*) adalah sebuah metode untuk membuat perangkat lunak menjadi tiga

¹<https://github.com/ftisunpar/BlueTape>

bagian: *Model*, *View*, dan *Controller*. *Model* adalah kelas yang merepresentasikan struktur data. *View* adalah informasi yang disajikan ke pengguna. *Controller* adalah penghubung antara *Model*, *View*, dan sumber daya lain yang dibutuhkan untuk mengolah HTTP *request* dan menghasilkan situs web.

3.1.1 Aturan Kontribusi BlueTape

Terdapat beberapa aturan apabila ingin berkontribusi pada pengembangan BlueTape. Aturan-aturan tersebut tertera pada file CONTRIBUTING.md (<https://github.com/ftisunpar/BlueTape/blob/master/CONTRIBUTING.md>).

3.1.1.1 Pengelompokan Module

Perangkat lunak BlueTape dikelompokkan dalam *module*. Setiap *module* memiliki nama yang mengikuti aturan *CamelCase*. Jika beberapa *module* tergabung pada satu topik yang sama, topik tersebut harus digunakan sebagai kata pertama dalam penamaan *module*. Contohnya: apabila nama topik adalah *Transkrip*, maka contoh nama *module*-nya adalah *TranskripRequest* dan *TranskripManage*.

Penamaan dokumen pada *controller*, *view*, *model*, config file, nama tabel, dan migration script menggunakan nama *module* atau topik. Contoh penamaan:

- *Controller*: controllers/TranskripRequest.php, controllers/TranskripManage.php
- *View*: views/TranskripRequest/*.php, views/TranskripManage/*.php
- *Model* (opsional): models/Transkrip/*_model.php
- *Config* file (opsional): config/Transkrip.php
- Nama tabel (opsional): Transkrip
- *Migration script* (opsional): migrations/20160222120000_Transkrip_initial.php

3.1.1.2 Model

Model dibuat hanya jika fungsi-fungsi di dalamnya digunakan lebih dari sekali. Apabila hanya digunakan sekali, letakkan fungsi pada *controller*.

3.1.1.3 Library bluetape

Library bluetape berisi fungsi-fungsi yang umum digunakan di BlueTape. Contoh: fungsi untuk konversi *email* ke NPM.

3.1.1.4 Hak Akses

Hak akses dan nama *module* diatur pada dokumen config/modules.php. Contoh:

```
$config['module-names'] = array(
    'TranskripRequest' => 'Permohonan Cetak Transkrip',
    'TranskripManage' => 'Manajemen Cetak Transkrip'
);

$config['modules'] = array(
    'TranskripRequest' => array('root', 'mahasiswa.ftis'),
    'TranskripManage' => array('root', 'tu.ftis')
);
```

```
$config['roles'] = array(
    'root' => 'pascal@unpar\\.ac\\.id',
    'tu.ftis' => '(shao\\.wei)@unpar\\.ac\\.id',
    'mahasiswa.ftis' => '7[123]\\d{5}@student\\.unpar\\.ac\\.id'
);
```

Apabila diperlukan, kontributor boleh menambahkan *role* baru pada *array config "roles"*. Setiap elemen *array* memetakan *role* dengan alamat *email* yang tergabung dalam *role* tersebut, dengan notasi *regular expression*.

3.1.2 Autentikasi

Setiap *module* wajib memeriksa hak akses sebelum ditampilkan. Hal tersebut dilakukan dengan cara memanfaatkan *template* berikut pada *controller*:

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class NamaPage extends CI_Controller {

    public function __construct() {
        parent::__construct();
        try {
            $this->Auth_model->checkModuleAllowed(get_class());
        } catch (Exception $ex) {
            $this->session->set_flashdata('error', $ex->getMessage());
            header('Location: /');
        }
    }

    // ... implementasikan method-method Anda yang lain di sini...
}
```

3.1.2.1 View

Setiap *view* menggunakan *template* yang menampilkan nama *module*, menu navigasi, dan *flash message* (jika diperlukan). Setiap *view* membutuhkan parameter *currentModule*, selain parameter-parameter lainnya. Jika ingin memanggil *view* dari *controller*, fungsi *get_class()* dapat digunakan. Berikut adalah cara sederhana memanggil *view*:

```
$this->load->view('NamaPage/main', array('currentModule' => get_class()));
```

View memanfaatkan *framework* Zurb Foundation, dan berisi *template* menu utama serta *framework*. Oleh karena itu, kode berikut digunakan untuk memulai membuat *view*:

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
?><!doctype html>
<html class="no-js" lang="en">
    <?php $this->load->view('templates/head_loggedin'); ?>
    <body>
        <?php $this->load->view('templates/topbar_loggedin'); ?>
        <?php $this->load->view('templates/flashmessage'); ?>
```

```
<!-- Tulislah isi view Anda di sini. -->

<script src="/public/foundation-6/js/vendor/jquery.min.js"></script>
<script src="/public/foundation-6/js/vendor/what-input.min.js"></script>
<script src="/public/foundation-6/js/foundation.min.js"></script>
<script src="/public/foundation-6/js/app.js"></script>
</body>
</html>
```

3.1.3 Fitur - Fitur BlueTape

Saat skripsi ini ditulis, BlueTape memiliki tiga layanan, yaitu Transkrip *Request / Manage*, Perubahan Kuliah *Request / Manage*, dan perekam jadwal dosen. Layanan Transkrip *Request / Manage* memberikan layanan untuk melakukan permohonan serta pencetakan transkrip mahasiswa. Layanan Perubahan Kuliah *Request / Manage* memberikan layanan untuk permohonan dan pencetakan perubahan jadwal kuliah oleh dosen. Layanan perekam jadwal dosen untuk merekam dan menampilkan jadwal dosen.

3.1.3.1 Transkrip *Request*

The screenshot shows the application's main menu at the top with links for Cetak Transkrip, Manajemen Cetak Transkrip, Perubahan Kuliah, Manajemen Perubahan Kuliah, Entri Jadwal Dosen, Lihat Jadwal Dosen, and Logout. Below the menu, there are two main sections:

- Permohonan Baru:** A form for submitting a new application. It includes fields for 'Yang memohon:' (Email: 7315029@student.unpar.ac.id), 'NPM:' (2015730029), 'Nama:' (ELLENA ANGELICA), 'Tipe Transkrip:' (DPS Bahasa Indonesia (Seluruh Semester)), and 'Keperluan:' (empty input field). A blue 'Kirim Permohonan' button is at the bottom.
- Histori Permohonan:** A table showing a single record of an application history. The columns are ID, Status, Tanggal Permohonan, Tipe Transkrip, Tanggal Jawab/Cetak, Keterangan, and Aksi. The data row is: ID (empty), Status (empty), Tanggal Permohonan (empty), Tipe Transkrip (DPS Bahasa Indonesia (Seluruh Semester)), Tanggal Jawab/Cetak (empty), Keterangan (empty), and Aksi (empty).

Gambar 3.2: Tampilan Cetak Transkrip

Gambar 3.2 menampilkan halaman utama saat cetak transkrip. Halaman ini hanya bisa diakses oleh *user* yang termasuk roles 'root' (admin) dan 'mahasiswa.ftis'. Terdapat form yang meminta *input* alamat *email* pemohon, npm, nama, tipe transkrip, dan keperluan. Input alamat *email* pemohon, npm, dan nama sudah terisi otomatis dan tidak bisa diubah lagi sehingga *user* hanya perlu mengganti tipe transkrip dan mengisi keperluan. Tipe transkrip memiliki tiga pilihan, yaitu: DPS Bahasa Indonesia (Seluruh Semester), DPS Bahasa Inggris (Seluruh Semester), dan LHS (Semester Terakhir). Selain form tersebut, terdapat tabel histori permohonan yang akan menampilkan riwayat permohonan cetak transkrip jika sudah pernah memohon. Tabel histori permohonan memiliki tujuh kolom, yaitu: ID, Status, Tanggal Permohonan, Tipe Transkrip, Tanggal Jawab/Cetak, Keterangan, dan Aksi (tindakan yang bisa dilakukan dengan record). Gambar 3.3 menampilkan tampilan setelah form permohonan transkrip baru dikirimkan. Pada Gambar 3.3 aksi yang tersedia hanya melihat detail permohonan.

Permintaan cetak transkrip sudah dikirim. Silahkan cek statusnya secara berkala di situs ini.

Permohonan Baru

Anda tidak bisa meminta cetak karena ada permintaan lain yang belum selesai.

Histori Permohonan

ID	Status	Tanggal Permohonan	Tipe Transkrip	Tanggal Jawab/Cetak	Keterangan	Aksi
#1	TUNGGU	Thursday, 15 November 2018	DPS_ID			

Gambar 3.3: Tampilan hasil Request Cetak Transkrip

3.1.3.2 Transkrip Manage

Permintaan Transkrip

Cari NPM: 2013730013

ID	Status	Tanggal Permohonan	Tipe Transkrip	NPM	Aksi
#1	MENUNGGU	Thursday, 15 November 2018	DPS_ID	2015730029	

Gambar 3.4: Tampilan Manajemen Transkrip BlueTape

Gambar 3.4 menampilkan tampilan halaman Manajemen Cetak Transkrip. Halaman ini hanya bisa diakses oleh *user* yang termasuk roles 'root' dan 'tu.ftis'. Halaman ini memiliki kolom pencarian yang dapat diisi dengan npm mahasiswa. Angka "2013730013" pada Gambar 3.4 merupakan placeholder saja, bukan *input user*. Apabila *input* kosong, maka semua permohonan akan ditampilkan pada tabel di bawah kolom pencarian. Pada Gambar 3.4, permohonan yang masuk baru satu saja. Apabila *input* diisi dan tombol "Cari ditekan", maka permohonan yang ditampilkan hanya permohonan milik npm yang diinput. User dapat melakukan empat aksi untuk tiap record yang ditampilkan: melihat detail permohonan (simbol mata), menolak permohonan (simbol jempol ke bawah), menyetujui permohonan (simbol print), dan menghapus permohonan (simbol tempat sampah).

3.1.3.3 Perubahan Kuliah Request

Permohonan Baru

Pemohon: 7315029@student.unpar.ac.id

Nama: ELLENA ANGELICA

Kode MK: _____ Nama Mata Kuliah: _____ Kelas: _____ Jenis Perubahan: Diganti

Dari Hari & Jam: _____ Dari Ruang: _____ Keterangan Tambahan: _____

Menjadi Hari & Jam: _____ Menjadi Ruang: _____

Kirim Permohonan **Tambah Pertemuan Ekstra**

Histori Permohonan

ID	Status	Tanggal Permohonan	Kode MK	Perubahan	Tanggal Jawab	Keterangan	Aksi
----	--------	--------------------	---------	-----------	---------------	------------	------

Gambar 3.5: Tampilan request perubahan kuliah

Gambar 3.5 menampilkan halaman Perubahan Kuliah. Halaman ini hanya bisa diakses oleh *user* yang termasuk roles 'root' dan 'staf.unpar'. Terdapat form yang meminta *input* alamat *email* pemohon, nama, kode mk, nama mata kuliah, kelas, jenis perubahan, hari dan jam sebelum dan sesudah diubah, serta ruang sebelum dan sesudah diubah. Input alamat *email* pemohon dan nama sudah terisi otomatis dan tidak bisa diubah lagi. Jenis perubahan memiliki tiga pilihan, yaitu: diganti, tambahan, dan ditadakan. Selain form tersebut, terdapat tabel histori permohonan yang akan menampilkan riwayat permohonan perubahan jadwal kuliah. Tabel histori permohonan memiliki delapan kolom, yaitu: ID, Status, Tanggal Permohonan, Kode MK, Perubahan, Tanggal Jawab, Keterangan, dan Aksi (tindakan yang bisa dilakukan dengan record).

3.1.3.4 Perubahan Kuliah Manage

Permohonan Perubahan Kuliah

ID	Status	Tanggal Permohonan	Kode MK	Perubahan	Aksi
#1	MENUNGGU	Thursday, 15 November 2018	AIF401	Ditiadakan	

Gambar 3.6: Tampilan manage perubahan kuliah

Gambar 3.6 menampilkan halaman Manajemen Perubahan Kuliah. Halaman ini hanya bisa diakses oleh *user* yang termasuk roles 'root' dan 'tu.ftis'. Halaman ini berisi riwayat permohonan perubahan kuliah. User dapat melakukan empat aksi untuk tiap record yang ditampilkan: melihat detail permohonan (simbol mata), menolak permohonan (simbol jempol ke bawah), menyetujui permohonan (simbol print), dan menghapus permohonan (simbol tempat sampah).

3.1.3.5 Entri Jadwal Dosen

Tambah Jadwal

Hari	Senin	Durasi	1 jam	Label	
Jam Mulai	7:00	Jenis	Konsultasi	Tambah	

Gambar 3.7: Tampilan tambah jadwal dosen

Gambar 3.7

Daftar Jadwal

	Senin	Selasa	Rabu	Kamis	Jumat
7-8	Bimbingan				
8-9					
9-10					
10-11					
11-12					
12-13					
13-14					
14-15					
15-16					
16-17					

Terakhir diupdate pada : 15 November 2018

Delete All **Eksport ke XLS**

Gambar 3.8: Tampilan jadwal dosen

Halaman Entri Jadwal Dosen hanya bisa diakses oleh *user* yang termasuk roles 'root' dan 'dosen.informatika'. Halaman ini terdiri dari dua bagian: form tambah jadwal dan daftar jadwal. Gambar 3.7 menunjukkan form tambah jadwal. Form ini meminta *input* hari, durasi, label, jam mulai dan jenis. Gambar 3.8 menunjukkan daftar jadwal milik *user*. Kolom yang memiliki isi bisa diklik untuk diedit. Gambar 3.9 menunjukkan tampilan edit jadwal.

Durasi	Label
Edit Jadwal	
Hari	
Senin	▼
Jam Mulai	▼
7:00	▼
Durasi	▼
1 jam	▼
Jenis	▼
Konsultasi	▼
Label	▼
Bimbingan	▼
Save	Delete

Gambar 3.9: Tampilan edit jadwal dosen

3.1.3.6 Lihat Jadwal Dosen

The screenshot shows a web-based application interface for viewing professor schedules. At the top, there's a navigation bar with icons for globe, print, and user, followed by the title 'Lihat Jadwal Dosen'. On the far right of the bar is a 'Logout' link. Below the title, there's a horizontal row of names: Thomas Anung Basuki, Chandra Wijaya, Cecilia Esti Nugraheni, Claudio Franciscus, Elisati Hulu, Husnul Hakim, Keenan Adewijaya Lemar, Kristopher David Hargjono, Luciana Abednego, Mariskha Tri Adithia, Natalia Natalia, Pascal Alfadian, Raymond Chandra Putra, and Rosa De Lima Endang Padmowati. A single name, Vania Natalia, is listed below this row. The main content area features a large table representing a weekly schedule grid. The columns are labeled with days of the week: Senin, Selasa, Rabu, Kamis, and Jumat. The rows represent time intervals from 7-8 to 16-17. A specific professor, identified by the initials 'LPM', has several slots filled with activities: 'Statistik untuk Komputasi' (Statistics for Computing) in the Rabu slot from 9-10 to 14-15, 'Penulisan Ilmiah' (Scientific Writing) in the Selasa slot from 10-11 to 15-16, 'Rapat' (Meeting) in the Senin slot from 11-12 to 16-17, and 'Pengantar Sistem Cerdas' (Introduction to Intelligent Systems) in the Kamis slot from 13-14 to 15-16. Some slots are empty or have other placeholder text like 'Rapat' or 'Statistik untuk Komputasi'. At the bottom left of the grid, there's a note: 'Terakhir diupdate pada : 27 Agustus 2018' and a blue button labeled 'Ekspor ke XLS'.

Gambar 3.10: Tampilan lihat jadwal dosen

Gambar 3.10 menampilkan halaman Lihat Jadwal Dosen. Halaman ini bisa diakses oleh *user* yang termasuk roles 'root', 'mahasiswa.informatika', dan 'dosen.informatika'. Halaman ini memiliki tab-tab yang masing-masing diberi label nama dosen. Di bawah tab terdapat tabel jadwal dari dosen yang tabnya aktif.

3.1.4 Hak Akses

Hak akses dan nama *module* diatur pada dokumen `config/modules.php` yang terletak di dalam direktori `config`. Hak akses dikelompokkan di dalam kelompok yang disebut *role*. Saat skripsi ini dibuat, hak akses dibagi ke dalam lima *role*: root, mahasiswa.ftis, tu.ftis, staf.unpar, dosen.informatika, dan mahasiswa.informatika. *Role* root berisi daftar alamat *email* dari pengembang bluetape. *Role* tu.ftis berisi daftar alamat *email* dari tata usaha ftis. *Role* mahasiswa.ftis berisi daftar alamat *email* dari mahasiswa ftis. *Role* staf.unpar berisi daftar alamat *email* dari staf unpar. *Role* dosen.informatika berisi daftar alamat *email* dari dosen informatika.

Setiap *role* memiliki batasan dalam mengakses *module* di BlueTape. *Role* root tidak memiliki batasan dan dapat mengakses setiap *module* yang ada. *Role* tu.ftis hanya dapat mengakses *module TranskripManage*, dan *module PerubahanKuliahManage*. *Role* mahasiswa.ftis hanya dapat mengakses *module TranskripRequest* dan *module LihatJadwalDosen*. *Role* staf.unpar hanya dapat mengakses *module PerubahanKuliahRequest*. *Role* dosen.informatika hanya dapat mengakses *module EntriJadwalDosen*.

3.1.5 Struktur Kelas BlueTape

BlueTape memiliki struktur kelas MVC (*Model View Controller*).

3.1.5.1 Model

Model yang sudah ada adalah:

- Auth_model: berisi algoritma yang dibutuhkan untuk autentikasi saat *login*.
- Email_model: berisi algoritma yang dibutuhkan untuk mengirim *email*.
- JadwalDosen_model: berisi algoritma yang dibutuhkan untuk fitur lihat jadwal dosen.
- PerubahanKuliah_model: berisi algoritma yang dibutuhkan untuk fitur perubahan kuliah *request* dan perubahan kuliah *manage*.
- Transkrip_model: berisi algoritma yang dibutuhkan untuk fitur transkrip *request* dan transkrip *manage*.

3.1.5.2 Controller

Controller yang sudah ada adalah:

- Auth: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat *login*.
- EntriJadwalDosen: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat memasukkan jadwal dosen.
- LihatJadwalDosen: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat *user* ingin melihat jadwal dosen.
- Migrate: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat melakukan *migration*.
- PerubahanKuliahManage: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat mengatur perubahan kuliah.
- PerubahanKuliahRequest: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat meminta perubahan kuliah.

- TranskripManage: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat mengatur permintaan transkrip.
- TranskripRequest: berfungsi untuk menjalankan perintah-perintah yang harus dijalankan saat *user* ingin meminta transkrip.

3.1.5.3 View

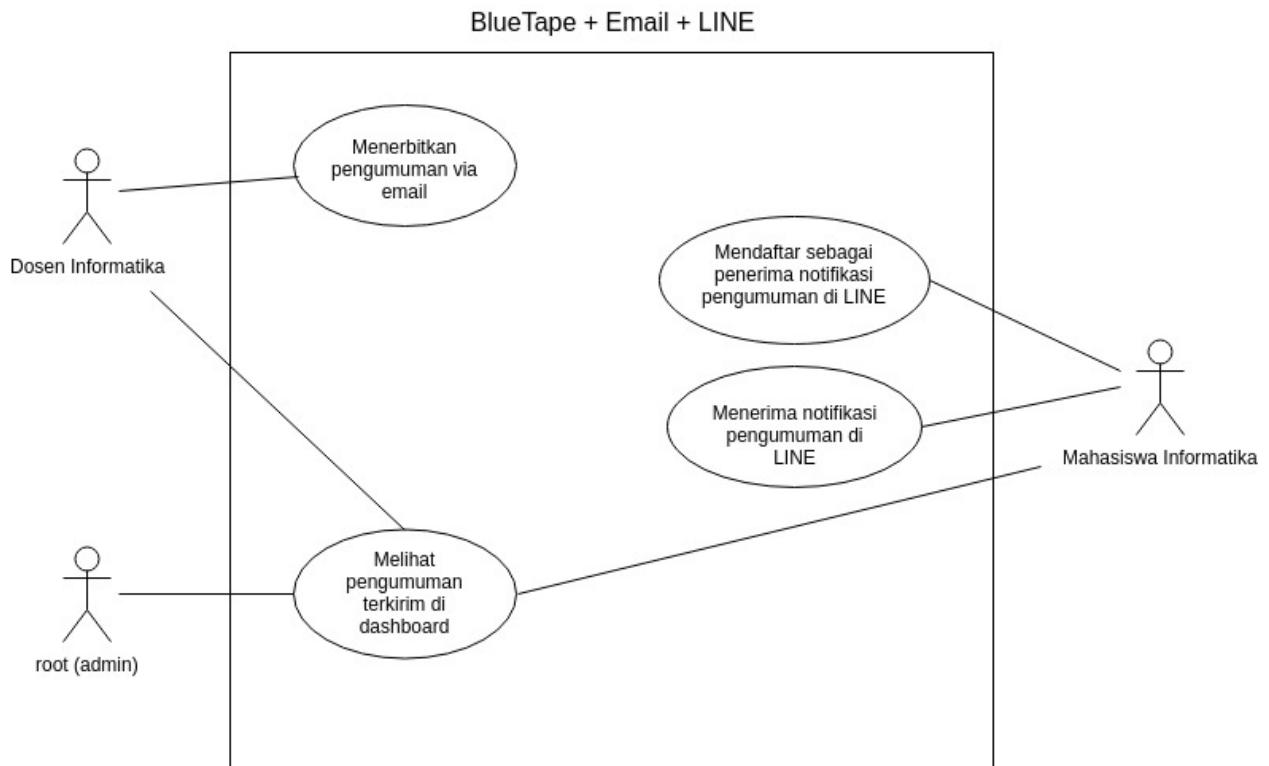
View yang sudah ada adalah:

- Auth/login: tampilan saat *login* (email, dan main), TranskripManage (email dan main), dan TranskripRequest (email dan main).
- EntriJadwalDosen/main: tampilan saat memasukkan jadwal dosen.
- LihatJadwalDosen/main: tampilan saat melihat jadwal dosen.
- PerubahanKuliahManage/email: tampilan *email* yang dikirimkan saat ada permintaan perubahan kuliah.
- PerubahanKuliahManage/main: tampilan saat di halaman pengolahan perubahan kuliah.
- PerubahanKuliahManage/printview: tampilan pengumuman perubahan kuliah.
- TranskripManage/email: tampilan *email* yang dikirimkan saat ada permintaan transkrip.
- TranskripManage/main: tampilan saat di halaman pengolahan permintaan transkrip.
- TranskripRequest/email: tampilan *email* yang dikirimkan saat permintaan transkrip diajukan.
- TranskripRequest/main: tampilan saat di halaman permintaan transkrip.

3.2 Analisis Sistem Usulan

Bagian ini membahas analisis fitur kolektor pengumuman.

3.2.1 Diagram *Use Case*



Gambar 3.11: Use case diagram fitur kolektor pengumuman

Gambar 3.11 merupakan gambar diagram *use case* fitur kolektor pengumuman. Pada diagram *use case* fitur kolektor pengumuman terdapat tiga aktor: dosen informatika, mahasiswa informatika, dan root (admin). Berikut ini adalah penjelasan dari skenario pada diagram *use case* tersebut:

1. Menerbitkan pengumuman via *email*

- Aktor: Dosen Informatika.
- Skenario Normal
 - (a) Dosen mengirimkan *email* ke alamat *email* yang dikhkususkan untuk menampung pengumuman di jurusan Teknik Informatika.
 - (b) BlueTape mengecek *email* tersebut pada periode tertentu.
 - (c) Jika alamat *email* yang dosen pakai terdaftar di BlueTape, maka BlueTape akan menampilkannya dan mengirim notifikasi ke LINE.
- Skenario Exception
 - (a) Dosen mengirimkan *email* ke alamat *email* yang dikhkususkan untuk menampung pengumuman di jurusan Teknik Informatika.
 - (b) BlueTape mengecek *email* tersebut pada periode tertentu.
 - (c) Jika alamat *email* yang dosen pakai tidak terdaftar di BlueTape, maka BlueTape akan mengabaikan *email* tersebut.

2. Mendaftar sebagai penerima notifikasi pengumuman di LINE

- Aktor: Mahasiswa Informatika.
- Skenario Normal

- (a) Mahasiswa mengikuti bot BlueTape dengan menambahkannya sebagai teman.
- (b) Bot BlueTape akan mengirim notifikasi kepada mahasiswa jika ada pengumuman baru di BlueTape.

3. Menerima notifikasi pengumuman di LINE

- Aktor: Mahasiswa Informatika.
- Skenario Normal
 - (a) Mahasiswa menerima notifikasi dari bot BlueTape saat ada pengumuman baru di BlueTape.
 - (b) Mahasiswa mengunjungi URL yang dicantumkan di pesan dari notifikasi tersebut.
 - (c) Mahasiswa perlu *login* menggunakan *email student* miliknya terlebih dahulu sebelum dapat mengunjungi URL tersebut.

4. Melihat pengumuman terkirim di *dashboard*

- Aktor: Dosen Informatika, Mahasiswa Informatika, dan root (admin).
- Skenario Normal
 - (a) Dosen Informatika, Mahasiswa Informatika, atau root (admin) mengunjungi menu Pengumuman. Menu Pengumuman berisi daftar pengumuman yang masuk ke BlueTape. Daftar tersebut disortir dari yang paling baru.
 - (b) Saat salah satu pengumuman diklik, maka isi dan detail pengumuman tersebut akan ditampilkan.

3.2.2 Analisis Heroku

3.2.2.1 Dependency

BlueTape membutuhkan *dependency* tambahan agar perangkat lunak dapat dijalankan. *Dependency* tambahan perangkat lunak BlueTape tertera pada dokumen `composer.json`. Berikut isinya:

```
{
  "require": {
    "google/apiclient": "^1.0",
    "ext-imap": "*",
    "phpoffice/phpexcel": "^1.8",
    "linecorp/line-bot-sdk": "^3.6"
  }
}
```

Pada saat skripsi ini ditulis, BlueTape telah memakai dua *package*: `package google/apiclient` dan `package phponline/phpexcel`. *Package* `google/apiclient` adalah *package* yang diperlukan untuk autentikasi akun saat masuk ke BlueTape. Sedangkan *package* `phponline/phpexcel` adalah *package* yang digunakan untuk menghasilkan dokumen excel. *Package* yang diperlukan untuk skripsi ini adalah *package* `ext-imap` dan `line-bot-sdk`. *Package* `ext-imap` digunakan untuk mengakses *email*. *Package* `line-bot-sdk` digunakan untuk menghubungkan perangkat lunak dengan layanan yang disediakan oleh LINE.

3.2.2.2 Process Type

BlueTape hanya membutuhkan satu *process type* di Procfile, yaitu *process type web* dan *process type release*. *Process type web* adalah *process type* yang digunakan untuk menerima arus HTTP eksternal dari router Heroku.

3.2.2.3 Procfile

Procfile adalah *file* yang menjelaskan bagian-bagian perangkat lunak yang dapat dieksekusi. Procfile berisi daftar *process type* beserta cara menjalankannya. BlueTape hanya memiliki satu *process type*, yaitu *process type web*. Isi Procfile adalah:

```
web: vendor/bin/heroku-php-apache2 www/
```

Maksud dari satu baris Procfile tersebut adalah: untuk menjalankan *process type* web, heroku harus menjalankan *server apache* di Heroku dan kemudian *server* menjalankan aplikasi web yang ada di direktori *www*. Perintah *vendor/bin/heroku-php-apache2* adalah perintah untuk menjalankan *server apache* di heroku yang ada di *package heroku-php-apache2*. *Package heroku-php-apache2* ini otomatis disediakan saat membuat perangkat lunak php di heroku sehingga tidak perlu ditambahkan di *composer.json*. Perintah *www/* berguna untuk mengarahkan *server apache* heroku ke direktori *www*.

3.2.2.4 Buildpack

Buildpack yang dipakai pada skripsi ini cukup hanya *heroku/php*. *Buildpack* ini secara otomatis akan dipakai oleh Heroku karena BlueTape memakai bahasa PHP.

3.2.2.5 Dyno

Jenis *dyno* yang dipakai pada skripsi ini adalah *free dyno*. Jumlah *dyno* hanya satu. *Dyno* tersebut merupakan *dyno* untuk *process type web*.

3.2.2.6 Config Vars

Environment variable yang perlu disimpan di *config vars* adalah konfigurasi *basis data*, konfigurasi autentikasi dengan OAuth Google, konfigurasi autentikasi *email* kolektor pengumuman, konfigurasi aplikasi, dan konfigurasi untuk terhubung ke LINE.

Berikut adalah nama-nama *config vars* yang akan dipakai pada skripsi ini beserta keterangannya:

- CI_DB_DATABASE: nama *database* yang digunakan.
- CI_DB_HOSTNAME: nama *host* dari *database* yang disebutkan di *config var CI_DB_DATABASE*.
- CI_DB_USERNAME: *username* yang digunakan untuk terhubung ke *database* yang disebutkan di *config var CI_DB_DATABASE*.
- CI_DB_PASSWORD: *password* dari *username* yang disebutkan di *config var CI_DB_USERNAME*.
- HEROKU_POSTGRESQL_BLUE_URL: URL *database*. Dibuat secara otomatis saat membuat *database*.
- GOOGLE_CLIENTID: Google Client ID, digunakan untuk melakukan autentikasi saat *user login*.
- GOOGLE_CLIENTSECRET: Google Client Secret, digunakan untuk melakukan autentikasi saat *user login*.
- ANNOUNCEMENT_EMAIL: alamat *email* yang dipakai untuk menampung pengumuman.
- ANNOUNCEMENT_PASSWORD: *password* untuk alamat *email* yang disebutkan di *config var ANNOUNCEMENT_EMAIL*.
- HOSTNAME_INCOMING_EMAIL: nama *host* dari alamat *email* yang disebutkan di *config var ANNOUNCEMENT_EMAIL*.

- CI_BASE_URL: *base URL* BlueTape.
- LINE_BOT_CHANNEL_SECRET: LINE Bot Channel Secret digunakan untuk terhubung ke *channel bot* untuk pengumuman.
- LINE_BOT_CHANNEL_TOKEN: LINE Bot Channel Token digunakan untuk terhubung ke *channel bot* untuk pengumuman.
- SMTP_HOST: SMTP *host*, konfigurasi untuk mengirim *email*.
- SMTP_PASS: SMTP *pass*, konfigurasi untuk mengirim *email*.
- SMTP_PORT: SMTP *port*, konfigurasi untuk mengirim *email*.
- SMTP_USER: SMTP *user*, konfigurasi untuk mengirim *email*.

3.2.2.7 Region

Region yang dipakai adalah *region default*, yaitu *United States*.

3.2.2.8 Stack

Stack yang dipakai adalah *stack* yang paling baru saat skripsi ini ditulis, yaitu heroku-18. *Stack* ini dipilih karena periode dukungannya paling lama, yaitu sampai bulan April 2023. Alasan lain adalah komputer lokal yang digunakan untuk mengerjakan skripsi ini menggunakan lingkungan yang mirip dengan Heroku, yaitu menggunakan Ubuntu 18.04.

3.2.2.9 Basis Data

Basis data yang digunakan untuk skripsi ini adalah basis data Heroku Postgres dengan plan *hobby-dev*. Alasan basis data ini digunakan adalah karena basis data ini adalah basis data yang secara otomatis disediakan oleh Heroku dan fiturnya cukup untuk digunakan oleh BlueTape.

Sebelumnya BlueTape menggunakan MySQL untuk basis datanya. Proses migrasi dari MySQL ke Heroku Postgres tidak rumit, karena menggunakan fitur *Migration* dari CodeIgniter. Namun, ada beberapa perubahan pada dokumen *Migration*. Perubahan-perubahan tersebut adalah:

- Menyelaraskan nama tabel karena sifat case sensitive pada Postgres
- Mengubah *Replace* menjadi *Insert* dan *Update* karena *Replace* tidak didukung oleh Postgres
- Mengubah tipe data kolom yang sebelumnya menggunakan DATETIME menjadi *timestamp*

3.2.3 Sinkronisasi Email

Awalnya sinkronisasi *email* akan dilakukan dengan memanfaatkan Gmail API. Namun, Gmail API membutuhkan *token* yang harus diperbarui tiap periode tertentu. Jika peneliti ingin *token* dapat diperbarui, maka *file refresh token* harus disimpan di Heroku. Namun, Heroku memiliki *filesystem* yang tidak mendukung penyimpanan *file* yang berubah-ubah. Sehingga, peneliti perlu mencari alternatif lain. Peneliti memutuskan menggunakan PHP IMAP untuk melakukan sinkronisasi *email*.

Sebelum melakukan sinkronisasi *email*, *email* khusus untuk menampung *email* pengumuman harus dibuat terlebih dahulu. *Email* dibuat melalui *provider email* Gmail. Setelah *email* selesai dibuat, fitur IMAP perlu dinyalakan terlebih dahulu.

Proses sinkronisasi *email* dimulai dengan membuat koneksi IMAP ke *email* pengumuman tersebut. Dengan menggunakan koneksi IMAP yang telah didapat, *email* difilter dengan mencari *email* yang belum dibaca saja. Apabila hasil pencarian tidak kosong, maka setiap *email* pada hasil pencarian akan diperiksa pengirimnya. Pengirim *email* akan dinyatakan sebagai pemberi

pengumuman yang sah apabila ia terdaftar di dalam daftar pengirim yang terverifikasi. Apabila sebuah *email* dinyatakan memiliki pengirim yang sah, maka informasi dari *email* tersebut akan diproses dan dimasukkan ke basis data.

Informasi yang perlu disimpan dari *email* pengumuman adalah alamat *email* pengirim, nama pengirim, tanggal *email* tersebut terkirim, subjek *email*, isi *email*, dan ketersediaan lampiran. Sebuah tabel baru diperlukan untuk menampung informasi ini. Tabel ini akan diakses saat halaman pengumuman akan ditampilkan. Setiap informasi dari *email* tersimpan, maka satu *push message* akan dikirim ke LINE.

Sinkronisasi *email* perlu dilakukan secara berkala dan otomatis. Pada skripsi ini sinkronisasi *email* dilakukan per hari dengan menggunakan *cron* dan *add-on* Heroku Scheduler.

3.2.4 Menghubungkan BlueTape dengan LINE

Produk LINE yang digunakan untuk menghubungkan BlueTape dengan LINE adalah LINE Messaging API. Produk ini paling memenuhi kriteria fitur kolektor pengumuman, yaitu dapat mengirimkan *push message*.

Sebelum menghubungkan BlueTape dengan LINE, ada beberapa hal yang harus dilakukan terlebih dahulu di LINE Developers *console*:

1. Membuat akun LINE dan mendaftar sebagai developer di LINE Developers *console*.
2. Membuat *provider* di LINE Developers *console*.
3. Membuat *channel* pada *provider* tersebut. *Channel* yang dibuat harus menggunakan *plan Developer Trial* karena *plan* ini yang memiliki fitur *push message*. Setelah *channel* terbuat, sebuah akun *bot* otomatis dibuat.

Pada *channel*, terdapat tiga bagian penting: *channel access token*, *channel secret*, dan *webhook URL*. *Channel access token* dan *channel secret* digunakan untuk autentikasi saat aplikasi BlueTape dan LINE berinteraksi. Kedua kode ini dapat diubah-ubah menggunakan tombol *issue* pada masing-masing kolom. Sedangkan *webhook URL* adalah alamat URL untuk menerima POST *request* berisi *event-event* yang terjadi pada *bot*. Contoh: *event following* yang terjadi saat ada akun LINE yang *follow* atau menambah akun *bot* sebagai teman. *Webhook URL* dapat diubah-ubah, namun harus menggunakan alamat https. Selain itu, fitur *csrf protection* milik Codeigniter harus dimatikan pada alamat URL tersebut.

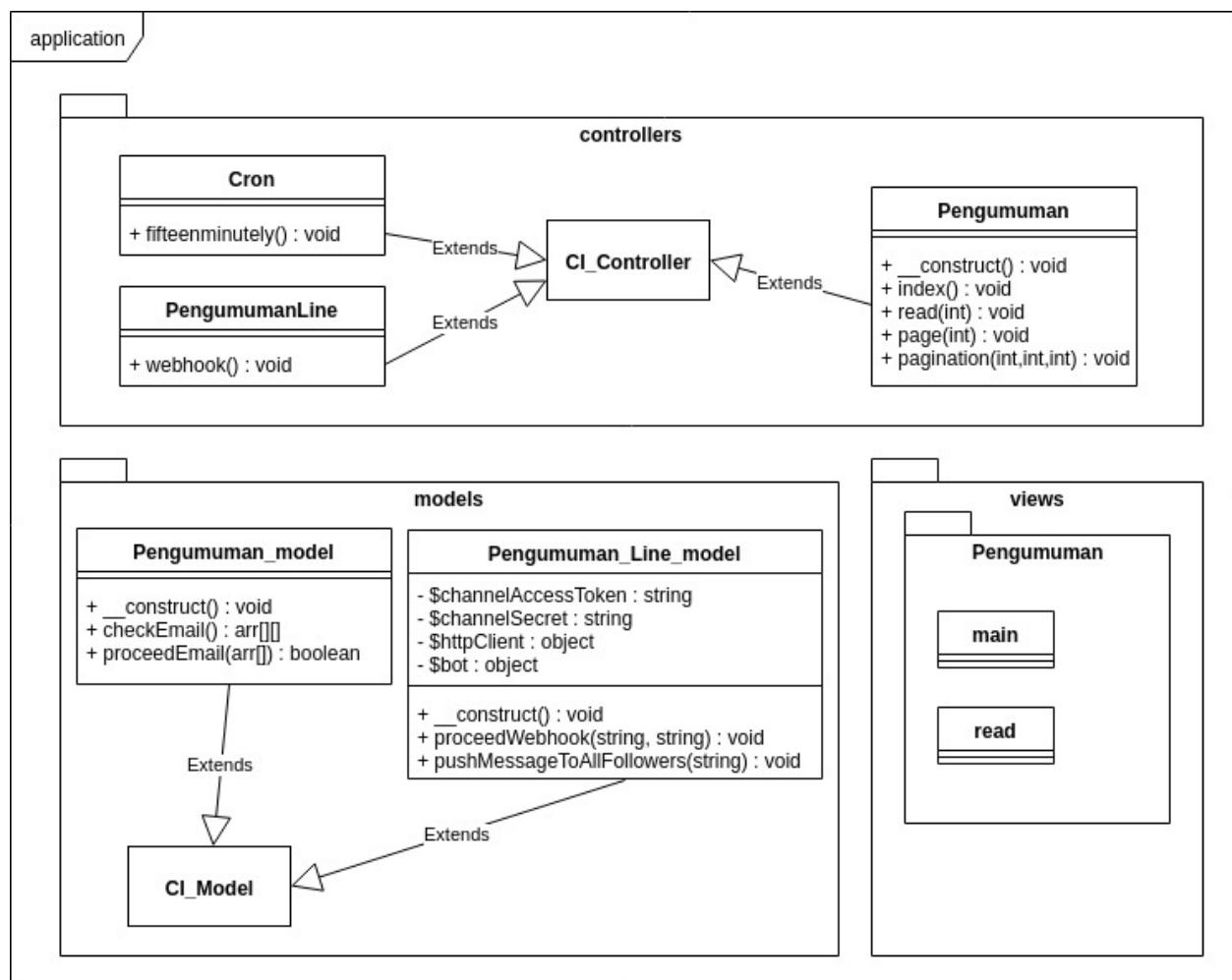
Pada skripsi ini, tidak semua *event* perlu ditangani. Event yang harus ditangani cukup *follow event* dan *unfollow event*. Saat *follow event* terjadi, *user id* dari akun *follower* akan disimpan di sebuah tabel di basis data. Apabila *user id* memblokir akun *bot* sehingga *unfollow event* terjadi, maka *user id* tersebut dihapus dari tabel. Penyimpanan ini diperlukan karena *user id* diperlukan saat mengirim *push message* dan tidak ada fungsi pada line-bot-sdk untuk mendapatkan *user id follower* akun *bot*.

BAB 4

PERANCANGAN

Bab ini membahas perancangan fitur kolektor pengumuman. Pembahasan dibagi menjadi tiga bagian, yaitu: perancangan kelas, perancangan basis data, dan perancangan antarmuka.

4.1 Perancangan Kelas



Gambar 4.1: Class diagram fitur kolektor pengumuman

Gambar 4.1 merupakan gambar diagram kelas yang dipakai untuk pembangunan fitur kolektor pengumuman. Penjelasan untuk diagram kelas tersebut akan diberikan di subbagian. Penjelasan dibagi menjadi tiga bagian: *controller*, *model*, dan *view*.

4.1.1 Model

Model yang digunakan ada dua, yaitu Pengumuman_model dan Pengumuman_Line_model.

4.1.1.1 Pengumuman_model

Pengumuman_model berisi algoritma yang dibutuhkan oleh fitur kolektor pengumuman. Pengumuman_model memiliki tiga *method*: __construct, checkEmail, dan proceedEmail. Tabel 4.1, 4.2, dan 4.3 menjelaskan secara rinci *method-method* tersebut.

Tabel 4.1: Rincian method __construct

Nama Method	__construct
Parameter Input	-
Parameter Output	-
Tabel yang berhubungan	-
Kelas yang berhubungan	-
Deskripsi	<i>Method</i> ini digunakan untuk konstruksi
Algoritma	<ol style="list-style-type: none"> 1. Konstruksi dari <i>parent</i>. 2. Load config auth dan modules.

Tabel 4.2: Rincian method checkEmail

Nama Method	checkEmail
Parameter Input	-
Parameter Output	<i>Array</i> dua dimensi. Dimensi pertama mewakili satu <i>email</i> , sedangkan dimensi kedua mewakili informasi dari <i>email</i> tersebut.
Tabel yang berhubungan	-
Kelas yang berhubungan	-
Deskripsi	
Algoritma	<ol style="list-style-type: none"> 1. Membuka koneksi IMAP ke <i>email</i> pengumuman. 2. Mencari <i>email</i> yang belum dibaca. 3. Apabila ada <i>email</i> yang belum dibaca, maka <i>email</i> tersebut dan informasinya dimasukkan ke dalam array. 4. Menutup koneksi IMAP dan mengembalikan array.

Tabel 4.3: Rincian method proceedEmail

Nama Method	proceedEmail
Parameter Input	Array asosiatif berisi informasi dari sebuah <i>email</i> .
Parameter Output	Mengembalikan <i>true</i> apabila <i>email</i> termasuk <i>email</i> pengumuman dan mengembalikan <i>false</i> apabila <i>email</i> tidak termasuk <i>email</i> pengumuman.
Tabel yang berhubungan	Pengumuman
Kelas yang berhubungan	Pengumuman_Line_model
Deskripsi	Method ini berfungsi untuk memroses <i>email</i> pengumuman
Algoritma	<ol style="list-style-type: none"> 1. <i>Load config</i> pengumuman 2. Memeriksa pengirim setiap <i>email</i>. Apabila pengirim terdaftar di <i>config</i> pengirimTerverifikasi, maka <i>email</i> tersebut diidentifikasi sebagai <i>email</i> pengumuman. 3. Informasi dari <i>email</i> pengumuman dimasukkan ke dalam baris <i>record</i> di tabel Pengumuman. 4. Pesan LINE@ dikirimkan dengan bantuan method pushMessageToAllFollowers yang terdapat pada Pengumuman_Line_model. Pesan tersebut berisi subjek dan pengirim <i>email</i> pengumuman. 5. Mengembalikan <i>true</i> apabila <i>email</i> diidentifikasi sebagai <i>email</i> pengumuman dan mengembalikan <i>false</i> apabila <i>email</i> diidentifikasi tidak termasuk <i>email</i> pengumuman.

4.1.1.2 Pengumuman_Line_model

Pengumuman_Line_model berisi algoritma yang dibutuhkan untuk berkomunikasi dengan Line API. Algoritma tersebut sengaja tidak disatukan ke dalam kelas Pengumuman_model karena menggunakan banyak *package* tambahan dan memiliki atribut-atribut yang tidak dibutuhkan semua *method* yang berada di Pengumuman_model. Atribut-atribut tersebut adalah: \$channelAccessToken, \$channelSecret, \$httpClient, dan \$bot. Pengumuman_Line_model memiliki tiga *method*: __construct, proceedWebhook, dan pushMessageToAllFollowers. Tabel 4.4, 4.5, dan 4.6 menjelaskan secara rinci *method-method* tersebut.

Tabel 4.4: Rincian method __construct

Nama Method	__construct
Parameter Input	-
Parameter Output	-
Tabel yang berhubungan	-
Kelas yang berhubungan	-
Deskripsi	Method ini digunakan untuk konstruksi
Algoritma	<ol style="list-style-type: none"> 1. Konstruksi dari <i>parent</i>. 2. <i>Load config auth</i> dan <i>modules</i>. 3. <i>Assign value</i> untuk setiap atribut.

Tabel 4.5: Rincian method proceedWebhook

Nama Method	proceedWebhook
Parameter Input	HTTP <i>request body</i> dan <i>X Line Signature</i>
Parameter Output	-
Tabel yang berhubungan	PengumumanLineFollowers
Kelas yang berhubungan	kelas-kelas di package LINE/LINEBot
Deskripsi	Method ini untuk memproses <i>event</i> yang masuk ke dalam method webhook yang terdapat di PengumumanLine
Algoritma	<ol style="list-style-type: none"> 1. Validasi signature dengan menggunakan method validateSignature yang membutuhkan parameter <i>input</i> HTTP <i>request body</i> dan <i>X Line Signature</i>. 2. Jika signature <i>valid</i>, maka jenis <i>event</i> yang masuk akan dicek dan ditangani. <i>Event</i> yang wajib ditangani adalah FollowEvent dan UnfollowEvent. Apabila FollowEvent terjadi (ada <i>user</i> yang mengikuti <i>bot</i> atau membuka blokir <i>bot</i>), maka id <i>user</i> LINE tersebut akan dimasukkan ke tabel PengumumanLineFollowers. Apabila UnfollowEvent terjadi (ada <i>user</i> yang memblokir <i>bot</i>), maka id <i>user</i> LINE tersebut akan dihapus dari tabel PengumumanLineFollowers. Penyimpanan id <i>user</i> diperlukan untuk mengirim pesan ke pengikut <i>bot</i>.

Tabel 4.6: Rincian method pushMessageToAllFollowers

Nama Method	pushMessageToAllFollowers
Parameter Input	Pesan yang ingin dikirimkan ke pengikut <i>bot</i>
Parameter Output	-
Tabel yang berhubungan	PengumumanLineFollowers
Kelas yang berhubungan	kelas-kelas di package LINE/LINEBot
Deskripsi	
Algoritma	<ol style="list-style-type: none"> 1. Membuat <i>query</i> untuk mendapatkan semua id <i>user</i> LINE pengikut <i>bot</i> dan memasukkan hasilnya ke suatu array. 2. Membuat objek baru dari kelas <i>text message builder</i> (kelas ini terdapat di LINE API) dengan parameter <i>input</i> pesan yang ingin dikirimkan ke pengikut <i>bot</i>. 3. Melakukan <i>multicast</i> dengan parameter <i>input</i> array id <i>user</i> LINE yang akan dikirimkan pesan dan <i>text message builder</i>.

4.1.2 View

View dibagi menjadi dua file php, yaitu *main* dan *read*. File *main* berfungsi untuk mengatur tampilan saat menampilkan daftar pengumuman. Sedangkan file *read* berfungsi untuk mengatur tampilan saat informasi dari satu *email* pengumuman ditampilkan.

4.1.3 Controller

Controller yang digunakan ada tiga, yaitu Cron, Pengumuman, dan PengumumanLine.

4.1.3.1 Cron

Controller Cron berfungsi untuk menjalankan perintah-perintah yang harus dijalankan pada jadwal tertentu. *Method* yang dimiliki hanya satu, yaitu: fifteenminutely(). Tabel 4.7 menjelaskan secara rinci method fifteenminutely().

Tabel 4.7: Rincian method fifteenminutely

Nama <i>Method</i>	fifteenminutely
Parameter <i>Input</i>	-
Parameter <i>Output</i>	-
Tabel yang berhubungan	-
Kelas yang berhubungan	Pengumuman_model
Deskripsi	<i>Method</i> yang perintah di dalamnya akan dijalankan setiap lima belas menit sekali. Untuk keperluan skripsi ini, method ini diisi dengan perintah untuk memeriksa <i>email</i> . Namun, apabila ada pengembangan lebih lanjut dan ada kebutuhan untuk menjalankan perintah setiap lima belas menit, maka isi <i>method</i> ini dapat ditambah.
Algoritma	<ol style="list-style-type: none"> Menjalankan method checkEmail() yang terdapat pada Pengumuman_model. Memeriksa <i>output</i> dari Pengumuman_model yang berupa array. Apabila tidak null, maka setiap elemen pada array tersebut akan dijadikan <i>input</i> dari method proceedEmail() yang terdapat pada Pengumuman_model. Menjalankan method proceedEmail() yang terdapat pada Pengumuman_model.

4.1.3.2 Pengumuman

Controller Pengumuman berfungsi untuk mengatur hubungan antara Pengumuman_model dan *view* yang ada di package Pengumuman. Pengumuman_model memiliki lima *method*, yaitu __construct, index, read, page, dan pagination. Tabel 4.8, 4.9, 4.10, 4.11, dan 4.12 menjelaskan secara rinci *method-method* tersebut.

Tabel 4.8: Rincian method __construct

Nama Method	__construct
Parameter Input	-
Parameter Output	-
Tabel yang berhubungan	-
Kelas yang berhubungan	-
Deskripsi	Method ini digunakan untuk konstruksi
Algoritma	<ol style="list-style-type: none"> 1. Konstruksi dari <i>parent</i>. 2. Mengecek <i>module</i> yang diizinkan. 3. <i>Load library</i> BlueTape 4. <i>Load model</i> Pengumuman_model 5. <i>Load database</i>

Tabel 4.9: Rincian method index

Nama Method	index
Parameter Input	-
Parameter Output	-
Tabel yang berhubungan	Pengumuman
Kelas yang berhubungan	-
Deskripsi	Mengatur halaman utama pengumuman
Algoritma	<ol style="list-style-type: none"> 1. Mengambil info <i>user</i> 2. Memasang <i>link</i> untuk setiap pengumuman yang ditampilkan pada daftar pengumuman. 3. Menampilkan halaman pertama.

Tabel 4.10: Rincian method read

Nama Method	read
Parameter Input	Id pengumuman
Parameter Output	-
Tabel yang berhubungan	Pengumuman
Kelas yang berhubungan	-
Deskripsi	Mengatur halaman yang menampilkan detail pengumuman
Algoritma	<ol style="list-style-type: none"> 1. Membuat <i>query</i> untuk mendapatkan seluruh informasi yang dimiliki oleh id pengumuman yang diinput dan memasukkannya ke suatu array. 2. Load <i>view</i> untuk <i>read</i> dan oper <i>array</i> tersebut ke <i>view</i>.

Tabel 4.11: Rincian method page

Nama Method	page
Parameter Input	nomor halaman yang ingin ditampilkan
Parameter Output	-
Tabel yang berhubungan	-
Kelas yang berhubungan	-
Deskripsi	Menampilkan halaman pengumuman pada nomor halaman yang diinput.
Algoritma	<ol style="list-style-type: none"> 1. Mengatur batas maksimal jumlah pengumuman yang ditampilkan pada satu halaman. 2. Memanggil method pagination dengan <i>input</i> nomor halaman, batas tersebut, dan id pengumuman yang akan ditampilkan di baris pertama.

Tabel 4.12: Rincian method pagination

Nama Method	pagination
Parameter Input	Nomor halaman yang ingin ditampilkan, batas maksimal jumlah pengumuman yang ditampilkan pada satu halaman, dan id pengumuman yang akan ditampilkan di baris pertama.
Parameter Output	-
Tabel yang berhubungan	Pengumuman
Kelas yang berhubungan	-
Deskripsi	Menampilkan halaman pengumuman pada nomor halaman yang diinput.
Algoritma	<ol style="list-style-type: none"> 1. Membuat <i>query</i> untuk mengetahui jumlah pengumuman yang ada di tabel Pengumuman 2. Membuat <i>query</i> untuk mendapatkan informasi dari pengumuman yang akan tampil di halaman tersebut dan memasukkannya ke suatu <i>array</i>. 3. Load <i>view</i> untuk main dan oper jumlah pengumuman yang ada di tabel Pengumuman, <i>array</i> tersebut, nomor halaman yang sedang ditampilkan, dan batas maksimal jumlah pengumuman yang ditampilkan pada satu halaman ke <i>view</i>.

4.1.3.3 PengumumanLine

Controller PengumumanLine berfungsi untuk menerima *webhook* dari LineAPI. PengumumanLine memiliki satu *method* yaitu *webhook*. *Method* ini sengaja dipisah dari *controller* Pengumuman karena proteksi csrf perlu dibuka untuk menerima webhook. Tabel 4.13 menjelaskan method tersebut secara rinci.

Tabel 4.13: Rincian method webhook

Nama Method	webhook
Parameter Input	Request berisi <i>event</i> dari Line API melalui POST
Parameter Output	-
Tabel yang berhubungan	-
Kelas yang berhubungan	Pengumuman_Line_model
Deskripsi	Method ini berfungsi untuk menerima request berisi <i>event</i> dari Line API melalui POST
Algoritma	<ol style="list-style-type: none"> 1. Memeriksa apakah <i>request</i> yang masuk melalui POST. Jika tidak, maka akan mengembalikan <i>http response code</i> 405. 2. Mengambil <i>request</i> yang masuk dan menyimpannya ke suatu variabel. 3. Memeriksa apakah ada HTTP_X_LINE_SIGNATURE pada POST yang masuk. Jika tidak ada, maka akan mengembalikan <i>http response code</i> 400. 4. Jika ada, maka jalankan method proceedWebhook dengan parameter <i>input</i> variabel yang berisi <i>request body</i> dan <i>X Line Signature</i>.

4.2 Perancangan Basis Data

Tabl-tabel yang sudah ada di BlueTape tidak diubah, namun ada dua tabel yang ditambahkan. Kedua tabel tersebut adalah tabel Pengumuman dan tabel Line_followers. Tabel pengumuman berguna untuk menyimpan informasi dari *email* pengumuman. Sedangkan tabel Line_followers berguna untuk menyimpan *user id* dari *follower* akun *bot* BlueTape.

4.2.1 Tabel Pengumuman

Rancangan Tabel Pengumuman dapat dilihat pada Tabel 4.14.

Tabel 4.14: Rancangan Tabel Pengumuman

Atribut	Tipe Data	Constraint	PK*	FK*
id	int	-	Ya	Tidak
namaPengirim	VARCHAR	256	Tidak	Tidak
emailPengirim	VARCHAR	256	Tidak	Tidak
waktuTerkirim	timestamp	-	Tidak	Tidak
subjek	VARCHAR	256	Tidak	Tidak
isi	TEXT	256	Tidak	Tidak
ketersediaanLampiran	VARCHAR	1	Tidak	Tidak

*PK = Primary Key

*FK = Foreign Key

Keterangan atribut:

- **id:** Id pengumuman. *Auto increment*.

- **namaPengirim:** Nama pengirim *email* pengumuman.
- **emailPengirim:** Alamat *email* pengirim *email* pengumuman.
- **waktuTerkirim:** Waktu terkirim *email* pengumuman.
- **subjek:** Subjek *email* pengumuman.
- **isi:** Isi *email* pengumuman. Boleh kosong.
- **ketersediaanLampiran:** Jika *email* pengumuman memiliki lampiran maka atribut ini memiliki *value* 'Y'. Jika tidak, maka *value*-nya 'N'.

4.2.2 Tabel PengumumanLineFollowers

Rancangan Tabel PengumumanLineFollowers dapat dilihat pada Tabel 4.15.

Tabel 4.15: Rancangan Tabel PengumumanLineFollowers

Atribut	Tipe Data	Constraint	PK*	FK*
userId	VARCHAR	256	Ya	Tidak

*PK = Primary Key

*FK = Foreign Key

Keterangan atribut:

- **userId:** *User id* dari akun LINE yang *follow* atau menambah akun *bot* BlueTape sebagai teman.

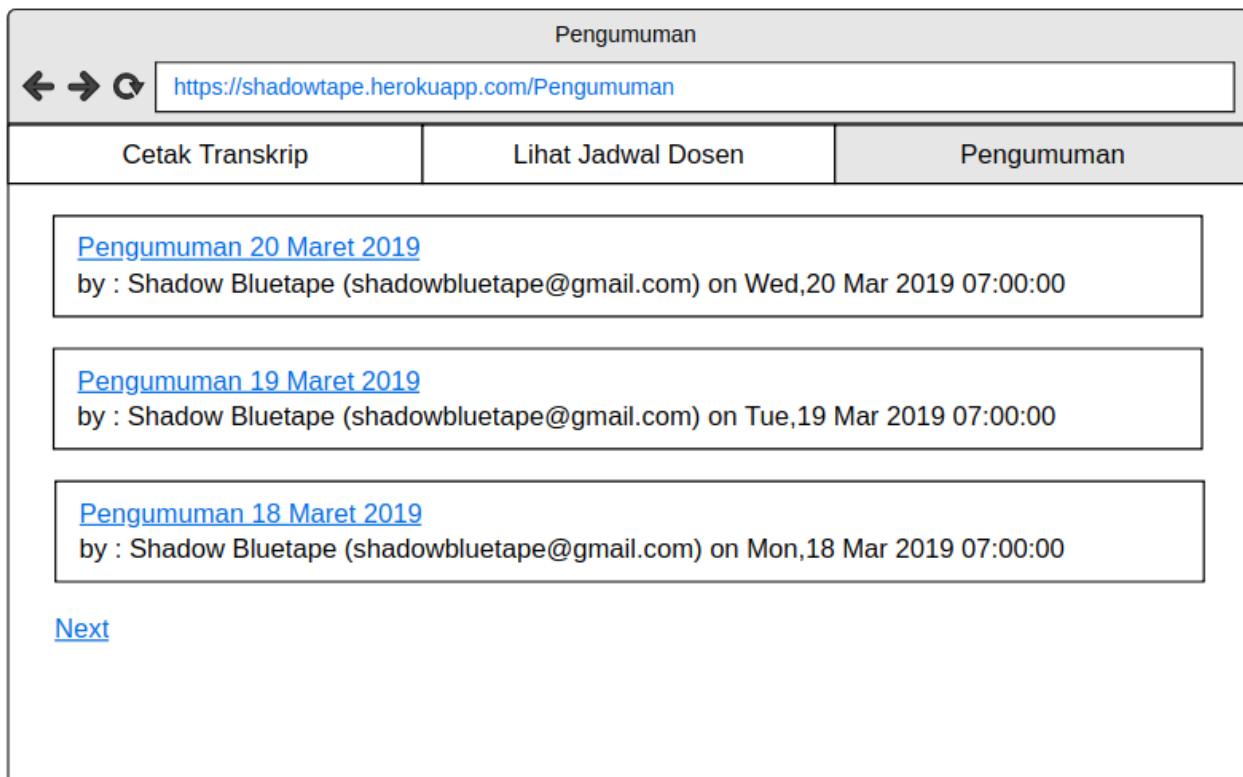
4.3 Perancangan Antarmuka

Bagian ini membahas perancangan antarmuka untuk fitur kolektor pengumuman pada BlueTape dan *bot* BlueTape.

4.3.1 Perancangan Antarmuka pada BlueTape

Antarmuka yang digunakan fitur kolektor pengumuman ada dua, yaitu *main* dan *read*. Penjelasan untuk desain kedua antarmuka tersebut terdapat di subbagian.

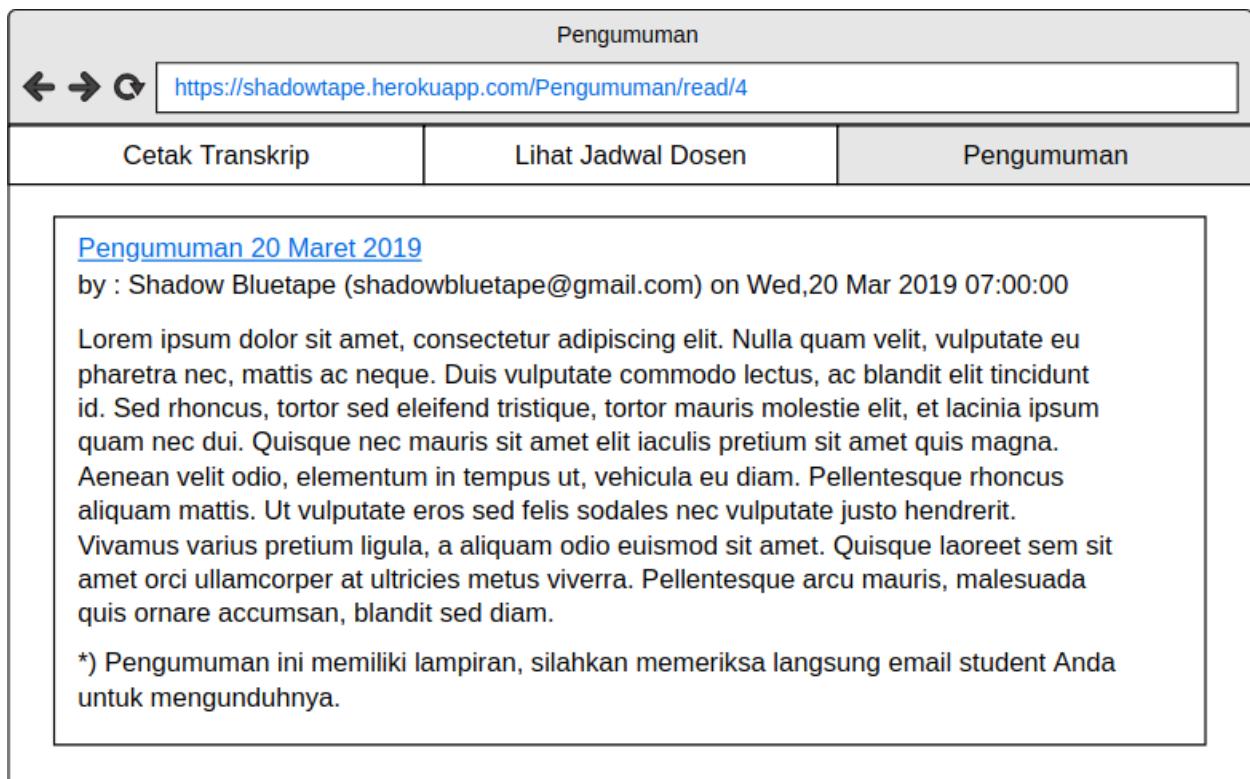
4.3.1.1 main



Gambar 4.2: *Mockup* antarmuka *main*

Antarmuka *main* menampilkan daftar pengumuman. Jumlah pengumuman yang ditampilkan di satu halaman dibatasi sebanyak 10 pengumuman. Di halaman *main* terdapat navigasi *next* dan *prev* untuk menampilkan daftar selanjutnya dan sesudahnya. Gambar 4.2 menampilkan *mockup* antarmuka *main*.

4.3.1.2 read



Gambar 4.3: *Mockup antarmuka read*

Antarmuka main menampilkan informasi detil dari pengumuman. Apabila pengumuman memiliki lampiran, maka tulisan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung email student Anda untuk mengunduhnya." akan muncul. Jika tidak ada, maka tulisan tersebut tidak akan muncul. Gambar 4.3 menampilkan *mockup* antarmuka *read*.

4.3.2 Perancangan Antarmuka pada Bot BlueTape



Gambar 4.4: Mockup antarmuka bot BlueTape

Gambar 4.4 menampilkan *mockup* antarmuka *bot* BlueTape. *Chat* pertama merupakan pesan yang akan ditampilkan saat *user* baru mengikuti *bot* atau membuka blokir *bot*. *Chat* kedua merupakan contoh pesan yang akan ditampilkan jika ada pengumuman baru yang masuk ke BlueTape. *Chat* terakhir merupakan pesan balasan jika *user* mengirimkan pesan ke *bot* dalam bentuk apapun (teks, *sticker*, gambar, video, dan suara).

BAB 5

IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas proses implementasi dan proses pengujian fitur kolektor pengumuman.

5.1 Implementasi

Bagian ini membahas implementasi dari perancangan yang telah dilakukan di Bab 4. Implementasi ini dilakukan dengan terlebih dahulu membuat akun Heroku lalu membuat aplikasi *web* baru di Heroku dan melakukan *deploy* ke dalamnya. Aplikasi *web* tersebut diberi nama Shadowtape dan dapat diakses di <https://shadowtape.herokuapp.com/>. Setelah itu, *source code* BlueTape dimodifikasi ulang sesuai hasil analisis arsitektur Heroku yang digunakan untuk BlueTape dan di-*deploy* lagi. Selain membuat akun Heroku, peneliti juga perlu membuat akun *email* baru dan akun LINE@ baru. Akun *email* baru menggunakan layanan *email* Gmail dengan alamat shadowbluetape@gmail.com. Akun LINE@ bernama Shadowtape.

5.1.1 Lingkungan Pengembangan

Berikut spesifikasi perangkat keras dan perangkat lunak yang dipakai untuk pengembangan pada skripsi ini:

Spesifikasi Perangkat keras

- Processor Intel® Celeron(R) CPU 1007U @ 1.50GHz x 2
- Graphics Intel® Ivybridge Mobile
- RAM 8 GB
- Harddisk 500GB SATA

Spesifikasi Perangkat lunak

- Sistem Operasi Ubuntu 18.04.1 LTS 64-bit
- Visual Studio Code version 1.31.0
- apache2 -v
- PHP 7.2.10 (cli)
- Composer version 1.7.2
- pgAdmin4 version 2.1 (Application Mode: Desktop)
- psql (PostgreSQL) 10.6
- heroku/7.21.0 linux-x64 node-v11.9.0

5.1.2 Implementasi Basis Data

Pada pembangunan fitur kolektor pengumuman, ada dua tabel yang ditambahkan. Kedua tabel tersebut adalah tabel Pengumuman dan tabel PengumumanLineFollowers. Pembuatan tabel menggunakan dua *file migration* terpisah: *file* 20181011103200_Pengumuman_Initial.php dan 20190210224400_PengumumanLineFollowers_initial.php.

5.1.3 Implementasi Kelas

Pada pembangunan fitur kolektor pengumuman, dibuat kelas-kelas berikut:

- Kelas *model* Pengumuman_model

Pengumuman_model merupakan kelas yang berisi algoritma yang dibutuhkan oleh fitur kolektor pengumuman. Kode program untuk kelas tersebut dapat dilihat di Listing 5.1.

Listing 5.1: /www/application/models/Pengumuman_model.php

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Pengumuman_model extends CI_Model {
    public function __construct() {
        parent::__construct();

        $this->load->config('auth');
        $this->load->config('modules');
    }

    public function checkEmail(){
        $newEmails = null;

        $hostname = getenv('HOSTNAME_INCOMING_EMAIL');
        $username = getenv('ANNOUNCEMENT_EMAIL');
        $password = getenv('ANNOUNCEMENT_PASSWORD');

        $inbox = imap_open($hostname,$username,$password) or die(
            'Cannot connect to Gmail: ' . imap_last_error());
        $emails = imap_search($inbox,'UNSEEN');

        if($emails) {
            $i = 0;
            foreach($emails as $emailNumber) {
                $invalid = false;
                $header = imap_headerinfo($inbox,$emailNumber);
                $from = isset($header->from) ? $header->from :
                    null;
                if($from != null){
                    $fromaddress = null;
                    foreach($from as $id => $object){
                        $fromaddress = isset($object->
                            mailbox) && isset($object->
                            host) ? $object->mailbox .
                            "@" . $object->host : null;
                }
            }
        }
    }
}
```

```
        }
        $bodymsg = '';
        $attachmentExist = 'N';
        $structure = imap_fetchstructure($inbox,
            ↪ $emailNumber);
        if(isset($structure->parts) && is_array(
            ↪ $structure->parts)) {
            if(isset($structure->parts[1])){
                $parts1 = $structure->
                    ↪ parts[1];
                if(isset($parts1->
                    ↪ disposition) &&
                    ↪ $parts1->disposition
                    ↪ == "ATTACHMENT"){
                    $attachmentExist =
                        ↪ 'Y';
                    $parts0 =
                        ↪ $structure->
                        ↪ parts[0];
                    if(isset($parts0->
                        ↪ parts[1])){
                        $bodymsg =
                            ↪ imap_qprint
                            ↪ (
                            ↪ imap_fetchbody
                            ↪ (
                            ↪ $inbox
                            ↪ ,
                            ↪ $emailNumber
                            ↪ ,
                            ↪ 1.2');
                            ↪ );
                }
            }
            else if(isset(
                ↪ $parts0->
                ↪ parts[0])){
                $bodymsg =
                    ↪ imap_qprint
                    ↪ (
                    ↪ imap_fetchbody
                    ↪ (
                    ↪ $inbox
                    ↪ ,
                    ↪ $emailNumber
                    ↪ ,
                    ↪ 1.1');
                    ↪ );
            }
        else{
            $bodymsg =
                ↪ imap_qprint
```

```

    ↵ (
    ↵   imap_fetchbody
    ↵   (
    ↵     $inbox
    ↵     ,
    ↵     $emailNumber
    ↵     ,
    ↵     '1'
    ↵   ));
    ↵ }
}
else{
    $bodymsg =
        ↵ imap_qprint(
        ↵   imap_fetchbody(
        ↵     ($inbox,
        ↵     $emailNumber,
        ↵     '2')));
    ↵ }
}
else{
    $bodymsg = imap_qprint(
        ↵   imap_fetchbody(
        ↵     $inbox, $emailNumber
        ↵     ,
        ↵     '1'));
    ↵ }
}
else{
    $bodymsg = imap_qprint(
        ↵   imap_fetchbody($inbox,
        ↵   $emailNumber, '1'));
    ↵ }

if($fromaddress != null){
    $newEmails[$i]['emailFrom'] =
        ↵ $fromaddress;
    $newEmails[$i]['from'] = isset(
        ↵ $header->fromaddress) ?
        ↵ $header->fromaddress :
        ↵ $fromaddress;
    if(isset($header->update)){
        $newEmails[$i]['date'] =
            ↵ date("Y-m-d\ H:i:s",
            ↵ $header->update);
    }
    else{
        $invalid = true;
    }

    if(isset($header->subject)){
        $newEmails[$i]['subject']
            ↵ = $header->subject;
    }
}

```

```
        }
        else{
            $invalid = true;
        }

        $newEmails[$i]['body'] = $bodymsg;
        ↪
        $newEmails[$i]['attachmentExist']
        ↪ = $attachmentExist;
    }

}

if($invalid){
    unset($newEmails[$i]);
}
else{
    $i++;
}

}

}

$errors = imap_errors();

imap_close($inbox);

return $newEmails;
}

public function proceedEmail($newEmail){
    $isPengumuman = false;
    $this->config->load('pengumuman');
    $terverifikasi = 0;
    $daftarEmailTerverifikasi = $this->config->item(
        ↪ pengirimTerverifikasi);
    foreach($daftarEmailTerverifikasi as $emailTerverifikasi){
        if($newEmail['emailFrom'] == $emailTerverifikasi){
            $terverifikasi = 1;
        }
    }

    if($terverifikasi == 1){
        $isPengumuman = true;
        $this->db->insert('Pengumuman', array(
            'namaPengirim' => $newEmail['from'],
            'emailPengirim' => $newEmail['emailFrom'],
            'waktuTerkirim' => $newEmail['date'],
            'subjek' => $newEmail['subject'],
            'isi' => $newEmail['body'],
            'ketersediaanLampiran' => $newEmail[
                ↪ attachmentExist']
        ));
        $justInserted = $this->db->select("*")->order_by('id', '
```

```

    ↪ desc")->limit(1)->get('Pengumuman')->row();
    $id = $justInserted->id;

    $this->load->model('Pengumuman_Line_model');
    $message = "Ada pengumuman baru dari " . $newEmail['
        ↪ from'] . ":" . $newEmail['subject'] . "'.".
        ↪ Silahkan klik link ini untuk melihatnya:" .
        ↪ base_url() . "pengumuman/read/" . $id;
    $this->Pengumuman_Line_model->pushMessageToAllFollowers
        ↪ ($message);
}
return $isPengumuman;
}
}

```

- Kelas *model* Pengumuman_Line_model

Pengumuman_Line_model merupakan kelas yang dikhususkan untuk algoritma untuk menghubungkan BlueTape dengan LINE API. Kode program untuk kelas tersebut dapat dilihat di Listing 5.2.

Listing 5.2: /www/application/models/Pengumuman_Line_model.php

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');

use LINE\LINEBot;
use LINE\LINEBot\HTTPClient;
use LINE\LINEBot\Event\AccountLinkEvent;
use LINE\LINEBot\Event\BeaconDetectionEvent;
use LINE\LINEBot\Event\FollowEvent;
use LINE\LINEBot\Event\JoinEvent;
use LINE\LINEBot\Event\LeaveEvent;
use LINE\LINEBot\Event\MessageEvent;
use LINE\LINEBot\Event\MessageEvent\AudioMessage;
use LINE\LINEBot\Event\MessageEvent\ImageMessage;
use LINE\LINEBot\Event\MessageEvent\LocationMessage;
use LINE\LINEBot\Event\MessageEvent\StickerMessage;
use LINE\LINEBot\Event\MessageEvent\TextMessage;
use LINE\LINEBot\Event\MessageEvent\UnknownMessage;
use LINE\LINEBot\Event\MessageEvent\VideoMessage;
use LINE\LINEBot\Event\PostbackEvent;
use LINE\LINEBot\Event\UnfollowEvent;
use LINE\LINEBot\Event\UnknownEvent;
use LINE\LINEBot\Exception\InvalidEventRequestException;
use LINE\LINEBot\Exception\InvalidSignatureException;

class Pengumuman_Line_model extends CI_Model {
    private $channelAccessToken;
    private $channelSecret;
    private $httpClient;
    private $bot;

    public function __construct(){

```

```
parent::__construct();

$this->load->config('auth');
$this->load->config('modules');

$this->channelAccessToken = getenv('LINE_BOT_CHANNEL_TOKEN');
$this->channelSecret = getenv('LINE_BOT_CHANNEL_SECRET');
$this->httpClient = new \LINE\LINEBot\HTTPClient\
    ↪ CurlHTTPClient($this->channelAccessToken);
$this->bot = new \LINE\LINEBot($this->httpClient, [
    ↪ channelSecret' => $this->channelSecret]);
}

public function proceedWebhook($httpRequestBody, $xLineSignature){
    $valid = $this->bot->validateSignature($httpRequestBody,
        ↪ $xLineSignature);
    if($valid){
        try{
            $events = $this->bot->parseEventRequest(
                ↪ $httpRequestBody, $xLineSignature);

            foreach ($events as $event) {
                if ($event instanceof MessageEvent) {
                    if ($event instanceof TextMessage
                        ↪ ) {
                        $this->bot->replyText(
                            ↪ $event->
                            ↪ getReplyToken(), "
                            ↪ Maaf, u saat ini bot
                            ↪ belum bisa membalas
                            ↪ pesan Anda.");
                    } elseif ($event instanceof
                        ↪ StickerMessage) {
                        $this->bot->replyText(
                            ↪ $event->
                            ↪ getReplyToken(), "
                            ↪ Maaf, u saat ini bot
                            ↪ belum bisa membalas
                            ↪ pesan Anda.");
                    } elseif ($event instanceof
                        ↪ LocationMessage) {
                        $this->bot->replyText(
                            ↪ $event->
                            ↪ getReplyToken(), "
                            ↪ Maaf, u saat ini bot
                            ↪ belum bisa membalas
                            ↪ pesan Anda.");
                    } elseif ($event instanceof
                        ↪ ImageMessage) {
                        $this->bot->replyText(
                            ↪ $event->
```

```

        ↪ getReplyToken(), "
        ↪ Maaf, usaatuiniubotu
        ↪ belumbisamembalas
        ↪ pesanAnda.");
    } elseif ($event instanceof
        ↪ AudioMessage) {
        $this->bot->replyText(
            ↪ $event->
            ↪ getReplyToken(), "
            ↪ Maaf, usaatuiniubotu
            ↪ belumbisamembalas
            ↪ pesanAnda.");
    } elseif ($event instanceof
        ↪ VideoMessage) {
        $this->bot->replyText(
            ↪ $event->
            ↪ getReplyToken(), "
            ↪ Maaf, usaatuiniubotu
            ↪ belumbisamembalas
            ↪ pesanAnda.");
    } elseif ($event instanceof
        ↪ UnknownMessage) {
        http_response_code(400);
        ↪ // Invalid event
        ↪ type
    } else {
        http_response_code(400);
        ↪ // Invalid event
        ↪ type
    }
} elseif ($event instanceof
    ↪ UnfollowEvent) {
    $id = $event->getUserId();
    $this->db->delete(
        ↪ PengumumanLineFollowers',
        ↪ array('userId' => $id));
} elseif ($event instanceof FollowEvent)
    ↪ {
    $this->db->insert(
        ↪ PengumumanLineFollowers',
        ↪ array(
            'userId' => $event->
                ↪ getUserId()
        )));
    $this->bot->replyText($event->
        ↪ getReplyToken(), 'Terimakasih telahmengikuti akunini.Pengumumanbaru diBluetape akandiberitahukan
        ↪ melaluiakunini.');
} elseif ($event instanceof JoinEvent) {

```

```

        // Not handled
    } elseif ($event instanceof LeaveEvent) {
        ↪
        // Not handled
    } elseif ($event instanceof
        ↪ PostbackEvent) {
        // Not handled
    } elseif ($event instanceof
        ↪ BeaconDetectionEvent) {
        // Not handled
    } elseif ($event instanceof
        ↪ AccountLinkEvent) {
        // Not handled
    } elseif ($event instanceof UnknownEvent)
        ↪ {
        http_response_code(400); //
        ↪ Invalid event type
    } else {
        http_response_code(400); //
        ↪ Invalid event type
    }
}
} catch (InvalidSignatureException $e) {
    http_response_code(400); // Invalid signature
} catch (InvalidEventRequestException $e) {
    http_response_code(400); // Invalid event
    ↪ request
}
}

public function pushMessageToAllFollowers($text){
    $tos = [];
    $query = $this->db->get('PengumumanLineFollowers');
    foreach ($query->result() as $row){
        $tos[] = $row->userId;
    }

    $ref = new ReflectionClass('LINE\LINEBot\MessageBuilder\
        ↪ TextMessageBuilder');
    $textMessageBuilder = $ref->newInstanceArgs(array_merge([$text]));
    $this->bot->multicast($tos, $textMessageBuilder);
}
}

```

- Kelas *controller Cron*

Cron merupakan kelas yang berfungsi untuk menjalankan perintah-perintah yang harus dijalankan pada jadwal tertentu. Pada skripsi ini perintah yang dijadwalkan adalah memeriksa *email*. Pada tahap perancangan, perintah untuk memeriksa *email* dijadwalkan tiap lima belas menit. Pada tahap implementasi, perintah tersebut dijadwalkan lebih jarang karena pemakaian *dyno free* dibatasi. Jadwal baru untuk perintah tersebut adalah setiap hari pada

pukul 12.00 siang. Kode program untuk kelas tersebut dapat dilihat di Listing 5.3.

Listing 5.3: /www/application/controllers/Cron.php

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Cron extends CI_Controller {

    public function daily() {
        try {
            $this->load->model('Pengumuman_model');
            $newEmails = $this->Pengumuman_model->checkEmail();
            $numberOfEmailsFound = 0;
            $numberOfAnnouncementEmails = 0;
            if($newEmails != null){
                foreach($newEmails as $newEmail){
                    $numberOfEmailsFound =
                        ↪ $numberOfEmailsFound + 1;
                    $isPengumuman = $this->Pengumuman_model->
                        ↪ proceedEmail($newEmail);
                    if($isPengumuman){
                        $numberOfAnnouncementEmails =
                            ↪ $numberOfAnnouncementEmails
                            ↪ + 1;
                    }
                }
            }
            log_message('info', "Successfully performed cron jobs. .
                ↪ The number of new emails found is .
                ↪ $numberOfEmailsFound . "and the number of new .
                ↪ emails that are announcements is .
                ↪ $numberOfAnnouncementEmails . .");
            http_response_code(200);
            echo json_encode([
                'message' => "Successfully performed cron jobs."
            ]);
        } catch (Exception $e) {
            log_message('error', $e->getMessage());
            http_response_code(500);
            echo json_encode([
                'message' => $e->getMessage()
            ]);
        }
    }
}
```

- Kelas *controller* Pengumuman

Pengumuman merupakan kelas yang berfungsi untuk mengatur hubungan antara Pengumuman_model dan view yang ada di package Pengumuman. Kode program untuk kelas tersebut dapat dilihat di Listing 5.4.

Listing 5.4: /www/application/controllers/Pengumuman.php

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Pengumuman extends CI_Controller {

    public function __construct() {
        parent::__construct();
        try {
            $this->load->helper('url');
            $this->session->set_userdata('redirect_url',
                ↪ current_url());
            $this->Auth_model->checkModuleAllowed(get_class());
        } catch (Exception $ex) {
            $this->session->set_flashdata('error', $ex->getMessage());
            header('Location:/');
        }
        $this->load->library('BlueTape');
        $this->load->model('Pengumuman_model');
        $this->load->database();
    }

    public function index() {
        // Retrieve logged in user data
        $userInfo = $this->Auth_model->getUserInfo();

        $this->db->select();
        $this->db->order_by('id', 'desc');
        $query = $this->db->get('Pengumuman');
        $announcements = $query->result_array();
        foreach ($announcements as &$announcement) {
            $announcement['url'] = "/pengumuman/read/" .
                ↪ $announcement['id'];
        }

        $this->page(1);
    }

    public function read($id){
        $this->db->where('id', $id);
        $this->db->select('*');
        $this->db->from('Pengumuman');
        $query = $this->db->get();
        $pengumuman= $query->row_array();
        if ($pengumuman === NULL) {
            show_404();
            exit;
        }
        $this->load->view('Pengumuman/read', array(
            'currentModule' => get_class(),
            'pengumuman' => $pengumuman
        )
    }
}
```

```

        );
    }

    public function page($page){
        $limit = 10;
        $this->pagination($page,$limit,((($page-1)*$limit));
    }

    public function pagination($page,$limit,$i){
        $jumlahPengumuman = $this->db->count_all('Pengumuman');
        $this->db->select('*');
        $this->db->order_by('waktuTerkirim', 'desc');
        $this->db->from('Pengumuman');
        $this->db->limit($limit,$i);
        $query = $this->db->get();
        $pengumumans = $query->result_array();
        $currentPage = $page;
        $pengumumanPerPage = $limit;
        $this->load->view('Pengumuman/main', array(
            'currentModule' => get_class(),
            'jumlahPengumuman' => $jumlahPengumuman,
            'pengumumans' => $pengumumans,
            'currentPage' => $currentPage,
            'pengumumanPerPage' => $pengumumanPerPage
        ));
    }
}

```

- Kelas *controller* PengumumanLine

Pengumuman_Line merupakan kelas yang berfungsi untuk menerima *webhook* dari Line API. Kode program untuk kelas tersebut dapat dilihat di Listing 5.5.

Listing 5.5: /www/application/controllers/PengumumanLine.php

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class PengumumanLine extends CI_Controller {

    public function webhook(){
        try{
            if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
                http_response_code(405);
                error_log('Method not allowed');
                exit();
            }

            $httpPostRequestBody = file_get_contents('php://input')
                ;
        }

        if (strlen($httpPostRequestBody) === 0) {
            http_response_code(400);
        }
    }
}

```

```

        error_log('Missing request body');
        exit();
    }

    $xLineSignature = $_SERVER['HTTP_X_LINE_SIGNATURE'];

    if (empty($xLineSignature)) {
        http_response_code(400); // Bad Request,
        ↪ Signature is Missing
    }
    else{
        $this->load->model('Pengumuman_Line_model');
        $this->Pengumuman_Line_model->proceedWebhook(
            ↪ $httpPostRequestBody, $xLineSignature);
    }
    http_response_code(200);
}
catch(Exception $e){
    http_response_code(500);
}
}
}

```

Untuk mendukung kinerja kelas-kelas tersebut, dibuat juga *file*:

- *File view main.php*.

File ini digunakan untuk mengatur tampilan halaman utama Pengumuman. Kode program untuk *file* tersebut dapat dilihat di Listing 5.6.

Listing 5.6: /www/application/views/Pengumuman/main.php

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');
?><!doctype html>
<html class="no-js" lang="en">
    <?php $this->load->view('templates/head_loggedin'); ?>
    <body>
        <?php $this->load->view('templates/topbar_loggedin'); ?>
        <?php $this->load->view('templates/flashmessage'); ?>
        <?php $this->load->view('templates/script.foundation'); ?>
        <div class="row">
            <div class="medium-12_column">
                <?php
                    if($jumlahPengumuman == 0){
                        echo "Tidak ada pengumuman";
                    }
                    else{
                        foreach($pengumumans as
                            ↪ $pengumuman):
                ?>
                        <div class="callout">

```

```

<h3><a href=<?=$
    ↪ pengumuman/
    ↪ read/" .
    ↪ $pengumuman['
    ↪ id']?> ><?=
    ↪ $pengumuman['
    ↪ subjek']?></
    ↪ a></h3>
<p>
    by : <?=
        ↪ $pengumuman
        ↪ ['
        ↪ namaPengirim
        ↪ ']?>
        ↪ (<?=
            ↪ $pengumuman
            ↪ ['
            ↪ emailPengirim
            ↪ ']?>)
        ↪ on
        ↪ <?=
            ↪ date(
            ↪ "D,d_
            ↪ M_ Y_ H:
            ↪ i:s",
            ↪
            ↪ strtotime
            ↪ (
            ↪ $pengumuman
            ↪ ['
            ↪ waktuTerkirim
            ↪ ']))?
        ↪ ?>
    </p>
</div>
<?php
    endforeach;
    if($currentPage>1){
        $previousPage =
            ↪ $currentPage-1;
        echo "<a href='/pengumuman
            ↪ /page-".
            ↪ $previousPage . "'>
            ↪ Previous</a>";
    }
    if($jumlahPengumuman >
        ↪ $currentPage*
        ↪ $pengumumanPerPage){
        $nextPage = $currentPage
            ↪ +1;
        echo "<a href='/pengumuman
            ↪ /page-".
            ↪ $nextPage . "'>
            ↪ Next</a>";
    }
}

```

```

        ↗ /page-". $nextPage .  

        ↗ "'>Next</a>";
    }  

}  

?>  

</div>  

</div>  

</body>  

</html>

```

- File view read.php.

File ini digunakan untuk mengatur tampilan halaman saat detail pengumuman ditampilkan. Kode program untuk file tersebut dapat dilihat di Listing 5.7.

Listing 5.7: /www/application/views/Pengumuman/read.php

```

<?php  

defined('BASEPATH') OR exit('No direct script access allowed');  

?><!doctype html>  

<html class="no-js" lang="en">  

<?php $this->load->view('templates/head_loggedin'); ?>  

<body>  

<?php $this->load->view('templates/topbar_loggedin'); ?>  

<?php $this->load->view('templates/flashmessage'); ?>  

<?php $this->load->view('templates/script.foundation'); ?>  

    <div class="row">  

    <div class="medium-12 column">  

        <div class="callout">  

            <h3><?= $pengumuman['subjek'] ?></h3>  

            <p>  

                by : <?= $pengumuman['  

                    ↗ namaPengirim'] ?> (<?=  

                    ↗ $pengumuman['emailPengirim'  

                    ↗ ] ?>) on <?= date("D, d M Y H:  

                    ↗ i:s", strtotime($pengumuman  

                    ↗ ['waktuTerkirim'])) ?>  

            </p>  

            <br><br>  

            <?php  

                if(!strpos($pengumuman['isi'], "<  

                    ↗ div")){  

            ?>  

                <?= nl2br(nl2br(  

                    ↗ $pengumuman['isi']))  

                    ↗ ; ?>  

            <?php  

            }  

            else{  

            ?>  

                <?= $pengumuman['isi']; ?>  

            <?php  

            }

```

```

?>
<br><br>
<?php if($pengumuman['
    ↪ ketersediaanLampiran'] == 'Y') :?>
    <p>
        *) Pengumuman ini
            ↪ memiliki
            ↪ lampiran,
            ↪ silahkan
            ↪ memeriksa
            ↪ langsung
            ↪ email
            ↪ student Anda
            ↪ untuk
            ↪ mengunduhnya.
            ↪
    </p>
    <?php
        endif;
    ?>
    </div>
</div>
</div>
</body>
</html>

```

- *File config pengumuman.php.*

File ini digunakan untuk menyimpan daftar pengirim pengumuman yang terverifikasi. Kode program untuk *file* tersebut dapat dilihat di Listing 5.8.

Listing 5.8: /www/application/config/pengumuman.php

```

<?php

defined('BASEPATH') OR exit('No direct script access allowed');

$config['pengirimTerverifikasi'] = array(
    'shadowbluetape@gmail.com', 'cheni@unpar.ac.id', 'mariskha@unpar.ac.
    ↪ id', 'anung@unpar.ac.id', 'moertini@unpar.ac.id', '
    ↪ natalia@unpar.ac.id', 'chandraw@unpar.ac.id', 'elisatih@unpar.
    ↪ ac.id', 'gkarya@unpar.ac.id', 'husnulhakim@unpar.ac.id', '
    ↪ joanna@unpar.ac.id', 'lionov@unpar.ac.id', 'luciana@unpar.ac.id
    ↪ ', 'claudio.franciscus@unpar.ac.id', 'pascal@unpar.ac.id', '
    ↪ rosad5@unpar.ac.id', 'vania.natali@unpar.ac.id', 'kristopher.
    ↪ h@unpar.ac.id', 'raymond.chandra@unpar.ac.id', 'keenan.
    ↪ leman@unpar.ac.id'
);

```

- *File migration 20181011103200_Pengumuman_Initial.php*

File ini digunakan untuk membuat tabel Pengumuman. Kode program untuk *file* tersebut dapat dilihat di Listing 5.9.

Listing 5.9: /www/application/migrations/20181011103200_Pengumuman_Initial.php

```
<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Migration_Pengumuman_initial extends CI_Migration {

    public function up() {
        $fields = array(
            'id' => array(
                'type' => 'int',
                'auto_increment' => TRUE
            ),
            'namaPengirim' => array(
                'type' => 'VARCHAR',
                'constraint' => '256'
            ),
            'emailPengirim' => array(
                'type' => 'VARCHAR',
                'constraint' => '256'
            ),
            'waktuTerkirim' => array(
                'type' => 'timestamp'
            ),
            'subjek' => array(
                'type' => 'VARCHAR',
                'constraint' => '256'
            ),
            'isi' => array(
                'type' => 'TEXT',
                'null' => TRUE
            ),
            'ketersediaanLampiran' => array(
                'type' => 'VARCHAR',
                'constraint' => '1'
            )
        );
        $this->dbforge->add_field($fields);
        $this->dbforge->add_key('id', TRUE);
        $this->dbforge->create_table('Pengumuman');
    }

    public function down() {
    }
}
```

- *File migration 20190210224400_PengumumanLineFollowers_initial.php.*

File ini digunakan untuk membuat tabel PengumumanLineFollowers. Kode program untuk file tersebut dapat dilihat di Listing 5.10.

Listing 5.10: /www/application/migrations/20190210224400_PengumumanLineFollowers_initial.php

```
<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Migration_PengumumanLineFollowers_initial extends CI_Migration {

    public function up() {
        $fields = array(
            'userId' => array(
                'type' => 'VARCHAR',
                'constraint' => '256'
            )
        );
        $this->dbforge->add_field($fields);
        $this->dbforge->add_key('userId', TRUE);
        $this->dbforge->create_table('PengumumanLineFollowers');
    }

    public function down() {

    }
}
```

Selain *file yang telah disebutkan*, ada beberapa *file* yang isinya harus diubah:

- *file config database.php*

Pada *file* ini, informasi yang berkaitan dengan basis data disesuaikan dengan informasi basis data yang dipakai di skripsi ini. Kode program untuk *file* tersebut dapat dilihat di Listing 5.11.

Listing 5.11: /www/application/config/database.php

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

/*
/ -----
/ DATABASE CONNECTIVITY SETTINGS
/ -----
/ This file will contain the settings needed to access your database.
/
/ For complete instructions please consult the 'Database Connection'
/ page of the User Guide.
/
/ -----
/ EXPLANATION OF VARIABLES
/ -----
/
/ ['dsn'] The full DSN string describe a connection to the database.
/ ['hostname'] The hostname of your database server.
/ ['username'] The username used to connect to the database
```

```
| ['password'] The password used to connect to the database
| ['database'] The name of the database you want to connect to
| ['dbdriver'] The database driver. e.g.: mysqli.
|   Currently supported:
|     cubrid, ibase, mssql, mysql, mysqli, oci8,
|     odbc, pdo, postgre, sqlite, sqlite3, sqlsrv
| ['dbprefix'] You can add an optional prefix, which will be added
|   to the table name when using the Query Builder class
| ['pconnect'] TRUE/FALSE - Whether to use a persistent connection
| ['db_debug'] TRUE/FALSE - Whether database errors should be displayed.
| ['cache_on'] TRUE/FALSE - Enables/disables query caching
| ['cachedir'] The path to the folder where cache files should be stored
| ['char_set'] The character set used in communicating with the database
| ['dbcollat'] The character collation used in communicating with the
|   ↵ database
|     NOTE: For MySQL and MySQLi databases, this setting is only used
|       as a backup if your server is running PHP < 5.2.3 or MySQL < 5.0.7
|       (and in table creation queries made with DB Forge).
|       There is an incompatibility in PHP with mysql_real_escape_string()
|       ↵ which
|       can make your site vulnerable to SQL injection if you are using a
|       multi-byte character set and are running versions lower than these.
|       Sites using Latin-1 or UTF-8 database character set and collation are
|       ↵ unaffected.
| ['swap_pre'] A default table prefix that should be swapped with the
|   ↵ dbprefix
| ['encrypt'] Whether or not to use an encrypted connection.
|
|   'mysql' (deprecated), 'sqlsrv' and 'pdo/sqlsrv' drivers accept TRUE/
|   ↵ FALSE
|   'mysqli' and 'pdo/mysql' drivers accept an array with the following
|   ↵ options:
|
|     'ssl_key' - Path to the private key file
|     'ssl_cert' - Path to the public key certificate file
|     'ssl_ca' - Path to the certificate authority file
|     'ssl_capath' - Path to a directory containing trusted CA certificates in
|       ↵ PEM format
|     'ssl_cipher' - List of *allowed* ciphers to be used for the encryption,
|       ↵ separated by colons (':')
|     'ssl_verify' - TRUE/FALSE; Whether verify the server certificate or not
|       ↵ ('mysqli' only)
|
| ['compress'] Whether or not to use client compression (MySQL only)
| ['stricton'] TRUE/FALSE - forces 'Strict Mode' connections
|   - good for ensuring strict SQL while developing
| ['ssl_options'] Used to set various SSL options that can be used when
|   ↵ making SSL connections.
| ['failover'] array - A array with 0 or more data for connections if the
|   ↵ main should fail.
| ['save_queries'] TRUE/FALSE - Whether to "save" all executed queries.
```

```

/*
 * NOTE: Disabling this will also effectively disable both
 * $this->db->last_query() and profiling of DB queries.
 * When you run a query, with this setting set to TRUE (default),
 * CodeIgniter will store the SQL statement for debugging purposes.
 * However, this may cause high memory usage, especially if you run
 * a lot of SQL queries ... disable this to avoid that problem.
 *
 * The $active_group variable lets you choose which connection group to
 * make active. By default there is only one group (the 'default' group).
 *
 * The $query_builder variables lets you determine whether or not to load
 * the query builder class.
 */
$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => getenv('CI_DB_HOSTNAME'),
    'username' => getenv('CI_DB_USERNAME'),
    'password' => getenv('CI_DB_PASSWORD'),
    'database' => getenv('CI_DB_DATABASE'),
    'dbdriver' => 'postgre',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);

```

- file config modules.php

Pada file ini, ditambahkan *modules* Pengumuman pada *config* 'modules'. Kode program untuk file tersebut dapat dilihat di Listing 5.12.

Listing 5.12: /www/application/config/modules.php.diff

```

diff --git a/www/application/config/modules.php b/www/application/config/
  ↳ modules.php
index 5dfb8b1..db58232 100644
--- a/www/application/config/modules.php
+++ b/www/application/config/modules.php
@@ -8,8 +8,8 @@ $config['module-names'] = array(
    'PerubahanKuliahRequest' => 'Perubahan_Kuliah',
    'PerubahanKuliahManage' => 'Manajemen_Perubahan_Kuliah',

```

```

'EntriJadwalDosen' => 'Entri_Jadwal_Dosen',
- 'LihatJadwalDosen' => 'Lihat_Jadwal_Dosen'
-
+ 'LihatJadwalDosen' => 'Lihat_Jadwal_Dosen',
+ 'Pengumuman' => 'Pengumuman'
);

$config['modules'] = array(
@@ -18,12 +18,13 @@ $config['modules'] = array(
    'PerubahanKuliahRequest' => array('root', 'staf.unpar'),
    'PerubahanKuliahManage' => array('root', 'tu.ftis'),
    'EntriJadwalDosen' => array('root', 'dosen.informatika' ),
- 'LihatJadwalDosen' => array('root', 'mahasiswa.informatika', 'dosen.
    ↲ informatika')
+ 'LihatJadwalDosen' => array('root', 'mahasiswa.informatika', 'dosen.
    ↲ informatika'),
+ 'Pengumuman' => array('root', 'mahasiswa.informatika', 'dosen.informatika')
    ↲
);
;

$config['roles'] = array(
- 'root' => array('pascal@unpar.ac.id', 'shao.wei@unpar.ac.id'),
- 'tu.ftis' => array('shao.wei@unpar.ac.id', 'pranyoto@unpar.ac.id', '
    ↲ walip@unpar.ac.id', 'dwina@unpar.ac.id'),
+ 'root' => array('pascal@unpar.ac.id', 'shao.wei@unpar.ac.id', '7315029
    ↲ @student.unpar.ac.id'),
+ 'tu.ftis' => array('pascal@unpar.ac.id'), // array('shao.wei@unpar.ac.id',
    ↲ 'pranyoto@unpar.ac.id', 'walip@unpar.ac.id', 'dwina@unpar.ac.id'),
    'mahasiswa.ftis' => '(7[123]\d{5})|(20[1-9][0-9]7[123][0-9]{4})
        ↲ |(61[678][0-9]{7})@student\\.unpar\\.ac\\\\.id',
    'staf.unpar' => '.+@unpar\\.ac\\\\.id',
    'dosen.informatika' => array ('cheni@unpar.ac.id', 'mariskha@unpar.ac.
        ↲ id', 'anung@unpar.ac.id', 'moertini@unpar.ac.id', '
        ↲ natalia@unpar.ac.id', 'chandraw@unpar.ac.id', 'elisatih@unpar.
        ↲ ac.id', 'gkarya@unpar.ac.id', 'husnulhakim@unpar.ac.id', '
        ↲ joanna@unpar.ac.id', 'lionov@unpar.ac.id', 'luciana@unpar.ac.id
        ↲ ', 'pascal@unpar.ac.id', 'rosad5@unpar.ac.id', 'vania.
        ↲ natali@unpar.ac.id', 'kristopher.h@unpar.ac.id', 'raymond.
        ↲ chandra@unpar.ac.id', 'keenan.leman@unpar.ac.id'),

```

- file config routes.php

Pada file ini ditambahkan routing berikut:

```
$route['pengumuman/page-(:num)'] = '/pengumuman/page/$1';
```

5.2 Pengujian

5.2.1 Lingkungan Pengujian

Berikut spesifikasi yang dipakai untuk pengujian pada skripsi ini:

- Heroku dengan spesifikasi :
 - *Region*: United States
 - *Stack*: heroku-18
 - *Framework*: PHP
 - Ukuran maksimum *slug*: 500 MiB
 - Heroku Git URL: [urlhttps://git.heroku.com/shadowtape.git](https://git.heroku.com/shadowtape.git)
 - *Buildpack*: heroku/php
 - *Domain*: <https://shadowtape.herokuapp.com/>
 - *Dyno Type*: Free Dynos
 - *Add-ons* Heroku Postgres dan Heroku Scheduler
- Akun *bot LINE@* dengan *plan Developer*

5.2.2 Pengujian Fungsional

Pengujian fungsional dilakukan dengan metode *black box testing*. Berikut adalah hasil pengujianya :

- Pengujian *Filter Email* Pengumuman

Pengujian ini bertujuan untuk menguji apakah *filter email* pengumuman berfungsi dengan baik. *Email* yang dikirim di pengujian ini memiliki subjek. Hasil pengujian dapat dilihat di Tabel 5.1.

Tabel 5.1: Pengujian Filter *Email* Pengumuman

Aksi	Reaksi yang diharapkan	Reaksi Perangkat Lunak
Mengirimkan <i>email</i> dengan <i>email</i> yang terdaftar lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman.
Mengirimkan <i>email</i> dengan <i>email</i> yang tidak terdaftar lalu menjalankan <i>Cron</i> .	<i>Email</i> tidak masuk ke basis data.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> tidak masuk ke basis data.

- Pengujian Mengirim *Email* dengan Isi *Email* yang Variatif

Pengujian ini bertujuan untuk menguji apakah isi *email* yang ditampilkan sesuai dengan yang diharapkan. Pengujian ini dilakukan dengan mengirimkan beberapa *email* dengan isi yang berbeda melalui salah satu *email* yang terdaftar di BlueTape, yaitu alamat *email* shadowbluetape@gmail.com. Hasil pengujian dapat dilihat di Tabel 5.2.

Tabel 5.2: Pengujian Mengirim *Email* dengan Isi *Email* yang Variatif

Aksi	Reaksi yang diharapkan	Reaksi Perangkat Lunak
------	------------------------	------------------------

Mengirimkan <i>email</i> tanpa subjek lalu menjalankan <i>Cron</i> .	<i>Email</i> tidak masuk ke basis data.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> tidak masuk ke basis data.
Mengirimkan <i>email</i> tanpa isi lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman.
Mengirimkan <i>email</i> dengan subjek dan isi lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman.
Mengirimkan <i>email</i> balasan lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman.
Mengirimkan <i>email</i> terusan lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman.
Mengirimkan <i>email</i> dengan lampiran lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman. Di bawah isi pengumuman, ada keterangan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung <i>email</i> student Anda untuk mengunduhnya.".	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman. Di bawah isi pengumuman, keterangan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung <i>email</i> student Anda untuk mengunduhnya." berhasil ditampilkan.
Mengirimkan <i>email</i> yang terdapat sisipan lampiran berupa gambar di isi <i>email</i> lalu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman. Di bawah isi pengumuman, ada keterangan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung <i>email</i> student Anda untuk mengunduhnya.". Isi pesan harus masih lengkap walaupun gambar tidak akan berhasil ditampilkan (karena file gambar tidak bisa disimpan di server).	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman. Di bawah isi pengumuman, keterangan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung <i>email</i> student Anda untuk mengunduhnya." berhasil ditampilkan. Isi pesan lengkap. Sesuai ekspektasi, gambar tidak bisa ditampilkan. Namun, ada keterangan alt yang berisi nama file.

Mengirimkan <i>email</i> yang isinya memakai berbagai jenis pemformatan yang bisa dilakukan di gmail, emoji yang disediakan gmail, dan sisipan URL. Setelah itu menjalankan <i>Cron</i> .	<i>Email</i> masuk ke basis data dan pengumuman ditampilkan di menu pengumuman. Isi <i>email</i> lengkap. Pemformatan tetap sama. Emoji tidak diharapkan bisa ditampilkan. Sisipan URL dapat ditampilkan dan URL dapat dikunjungi.	Reaksi sesuai dengan yang diharapkan. <i>Email</i> masuk ke basis data dan pengumuman dapat dilihat di menu pengumuman. Isi <i>email</i> lengkap. Pemformatan tetap sama. Emoji dapat ditampilkan. Beberapa emoji berubah bentuk tapi tetap memiliki bentuk yang sama. Beberapa emoji persis sama dengan yang ada di isi <i>email</i> yang asli. Sisipan URL dapat ditampilkan dan URL dapat dikunjungi.
---	--	--

- Pengujian Notifikasi LINE

Pengujian ini bertujuan untuk menguji apakah notifikasi LINE dapat terkirim ke bot LINE. Tabel 5.3.

Tabel 5.3: Pengujian Notifikasi LINE

Aksi	Reaksi yang diharapkan	Reaksi Perangkat Lunak
Follow akun bot.	User Id LINE tercatat di basis data.	Reaksi sesuai dengan yang diharapkan. User Id LINE berhasil tercatat di basis data.
Mengirimkan <i>email</i> dengan <i>email</i> yang terdaftar lalu menjalankan <i>Cron</i> . <i>Email</i> harus memiliki subjek.	Setelah <i>Cron</i> sukses dijalankan, notifikasi LINE dari akun bot muncul.	Reaksi sesuai dengan yang diharapkan. Notifikasi LINE dari akun bot muncul.
Membuka pesan LINE yang masuk. Setelah itu membuka URL yang tercantum di pesan.	URL dapat dibuka. Apabila belum <i>login</i> ke BlueTape, pengguna akan diarahkan ke menu <i>login</i> . Setelah <i>login</i> , pengguna diarahkan kembali ke URL tersebut.	Reaksi sesuai dengan yang diharapkan. URL dapat dibuka. Apabila belum <i>login</i> ke BlueTape, pengguna akan diarahkan ke menu <i>login</i> . Setelah <i>login</i> , pengguna diarahkan kembali ke URL tersebut.

5.2.3 Pengujian Eksperimental

Untuk Mahasiswa
Rekan-rekan mahasiswa,
Terikait skripsi salah satu bimbingan saya, **Ellena / Kolektor Pengumuman Informatika**, saya memohon bantuan untuk melakukan pengujian eksperimental. Caranya, dengan add friend @ibz3613t atau scan QR code di bawah ini menggunakan aplikasi LINE.

Saat ada pengumuman dari saya / dosen lain, maka Anda akan menerima notifikasi di aplikasi LINE Anda.
Supaya menarik, kisi-kisi UAS akan saya kirimkan melalui aplikasi ini.
Nantinya, Anda akan diminta untuk mengisi kuisioner di alamat: <https://forms.gle/e7awCboE94kLEVlZ9>

Untuk Dosen
Kirimkan e-mail ke shadowbluetape@gmail.com untuk memberikan pengumuman kepada mahasiswa dari akun e-mail @unpar.ac.id. Beberapa catatan:

1. Untuk saat ini, mohon gunakan web-based GMail. Jika menggunakan mail client sepertinya masih ada bug, konten kosong.
2. Karena alasan biaya, pengumuman akan dikirimkan ke LINE setiap jam 12 siang (tidak instan)
3. Attachment tidak akan muncul, tidak disimpan di server

Nantinya, Anda akan diminta untuk mengisi kuisioner di alamat <https://forms.gle/e7awCboE94kLEVlZ9>

Gambar 5.1: Pengumuman Ujian Eksperimental

Pengujian eksperimental untuk aplikasi ini dilakukan pada mata kuliah AIF182100 (Analisis dan Desain Perangkat Lunak) kelas B dimana dosen kelas tersebut membuat pengumuman kepada mahasiswa untuk melakukan *add friend* dan menunggu pengumuman dari dosen. Pengumuman dapat dilihat pada Gambar 5.1. Berikut tata cara pengujianya:

1. Mahasiswa menambahkan bot Shadowtape dengan mencari id LINE "@ibz3613t" atau memindai kode QR (Gambar 5.2).

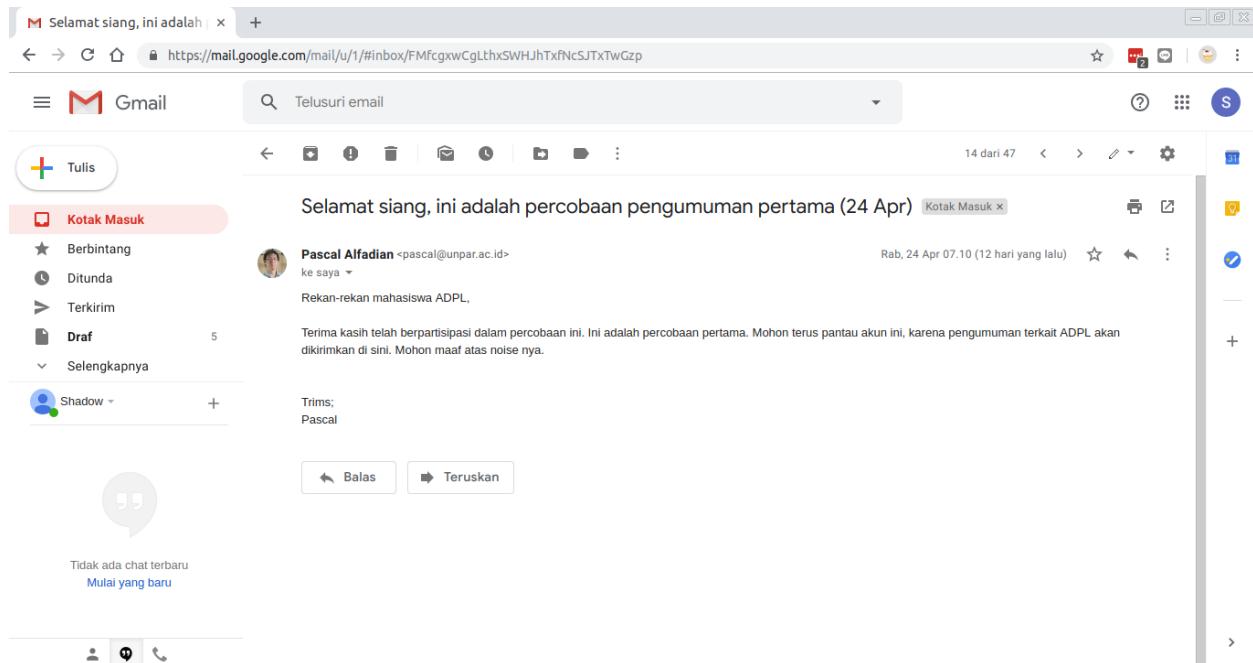


Gambar 5.2: Kode QR Bot Shadowtape

2. Dosen mengirimkan *email* berisi pengumuman ke shadowbluetape@gmail.com menggunakan alamat *email* yang terdaftar di BlueTape, yaitu alamat *email* dengan domain unpar.
3. Apabila dosen mengirimkan *email*, maka notifikasi LINE dari akun Shadowtape akan diterima mahasiswa pada pukul 12.00 siang setelah *email* tersebut dikirim.
4. Apabila mahasiswa menerima notifikasi tersebut, mahasiswa akan menerima pesan yang terdapat subjek *email* dan URL di dalamnya. URL tersebut mengarah ke halaman detail pengumuman pada domain <http://shadowtape.herokuapp.com/>.

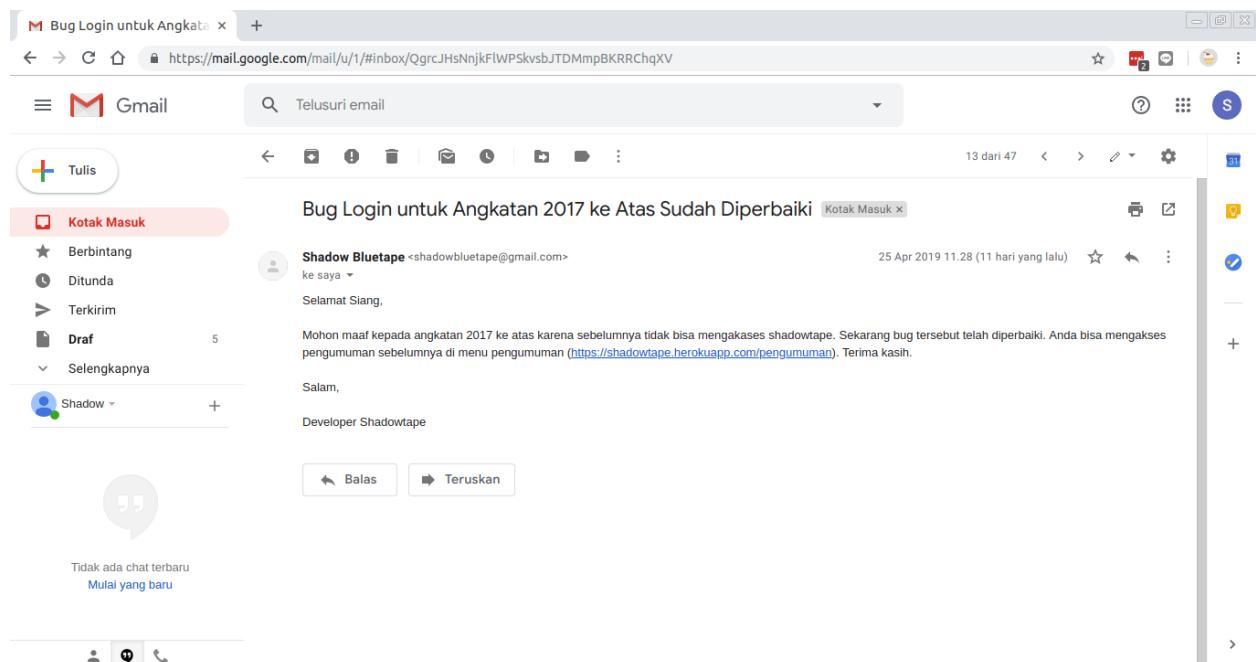
5. Apabila mahasiswa mengklik URL tersebut dan belum *login*, mahasiswa akan diarahkan ke halaman *login* sebelum diarahkan kembali ke URL tersebut.
6. Setelah masa pengujian berakhir, dosen dan mahasiswa mengisi kuesioner pada URL : <https://forms.gle/e7awCboE94kLEVLZ9>.

5.2.3.1 Pelaksanaan Pengujian



Gambar 5.3: *Email* pada tanggal 24 April 2019 oleh Bapak Pascal

Pada tanggal 24 April 2019 pukul 07.10, Bapak Pascal Alfadian sebagai dosen mata kuliah AIF182100 (Analisis dan Desain Perangkat Lunak) kelas B mengirimkan *email* ke alamat *email* shadowbluetape@gmail.com. Isi *email* dapat dilihat pada Gambar 5.3.



Gambar 5.4: *Email* pada tanggal 25 April 2019 oleh peneliti

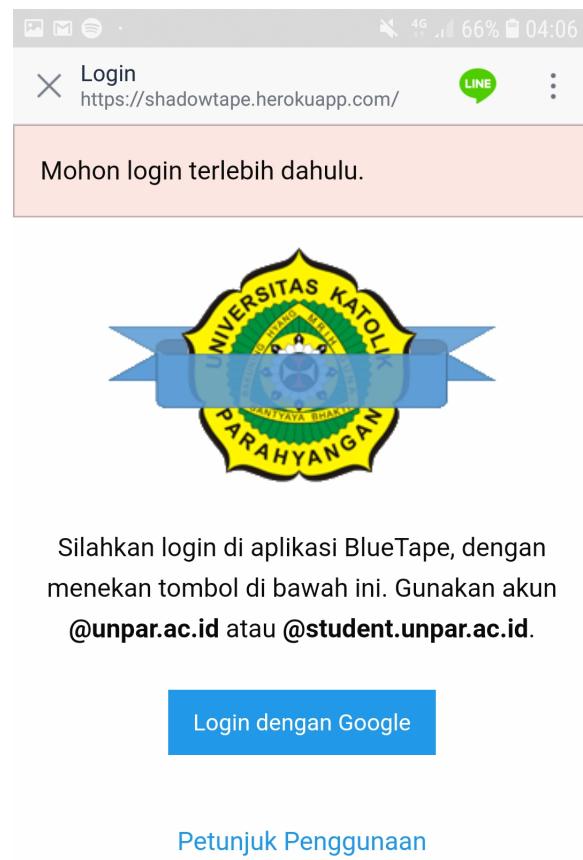
Pada tanggal 25 April 2019, peneliti menerima laporan bahwa ada masalah *login* untuk angkatan 2017 ke atas. Peneliti menemukan masalah tersebut terjadi karena *repository* skripsi ini ((<https://github.com/EllenaAngelica/BlueTape>)) tertinggal 4 *commit* dari *repository* asal (<https://github.com/ftisunpar/BlueTape>). Peneliti memperbaiki masalah tersebut dengan melakukan `git pull`. Pada hari tersebut pukul 11.28, peneliti memakai *email* shadowbluetape@gmail.com ke alamat *email* ke shadowbluetape@gmail.com untuk memberitahu masalah tersebut telah diperbaiki sekaligus melakukan percobaan kedua. Isi *email* dapat dilihat pada Gambar 5.4.

5.2.3.2 Hasil Pengujian



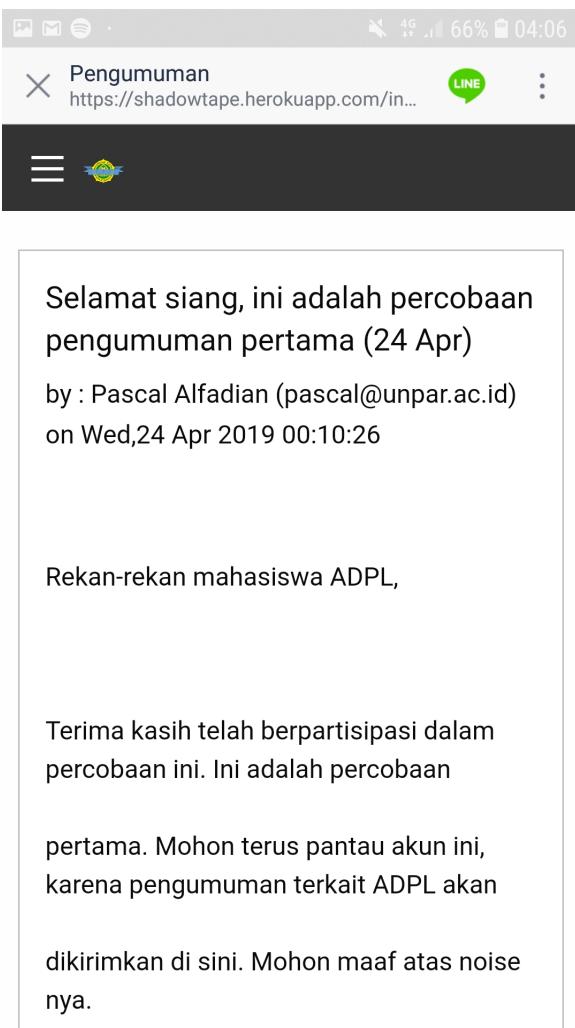
Gambar 5.5: Tampilan pesan LINE setelah pengumuman disebar

Gambar 5.5 menampilkan tampilan pesan LINE yang masuk setelah pengumuman disebar. Karena peneliti tidak sengaja memblokir bot Shadowtape dan tidak dapat membuka blokirnya, tampilan pada gambar tersebut memiliki tanda "Diblokir".



Gambar 5.6: Tampilan BlueTape setelah URL dibuka dan pengguna belum *login*

Gambar 5.6 menampilkan tampilan BlueTape setelah URL dibuka dan pengguna belum *login*.



Gambar 5.7: Tampilan URL pengumuman yang diumumkan oleh Pak Pascal Alfadian

Gambar 5.7 menampilkan tampilan URL pengumuman yang diumumkan oleh Pak Pascal Alfadian.



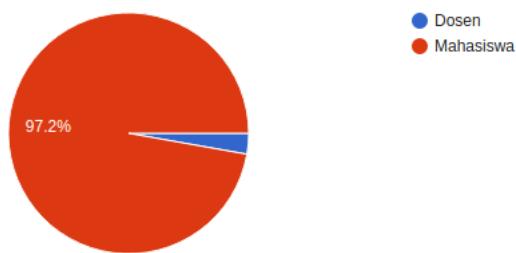
Gambar 5.8: Tampilan URL pengumuman yang diumumkan oleh peneliti

Gambar 5.8 menampilkan tampilan URL pengumuman yang diumumkan oleh peneliti.

5.2.3.3 Hasil Kuesioner

Apakah Anda dosen atau mahasiswa?

72 responses

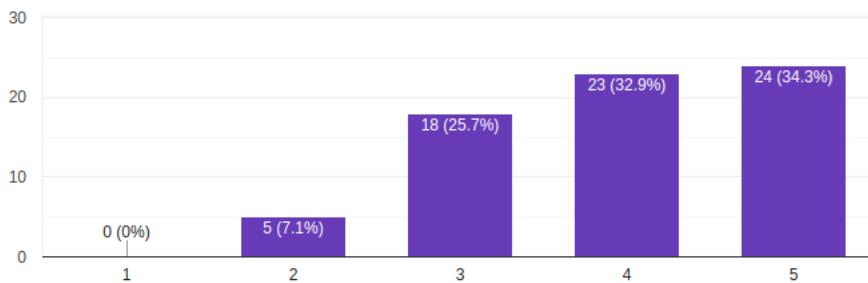


Gambar 5.9: Diagram profil responden

Setelah masa pengujian berakhir, responden yang mengisi kuesioner adalah 72 responden yang terdiri dari 70 mahasiswa dan 2 dosen. Diagram profil responden dapat dilihat pada Gambar 5.9.

Seberapa sering Anda membuka email student Anda?

70 responses

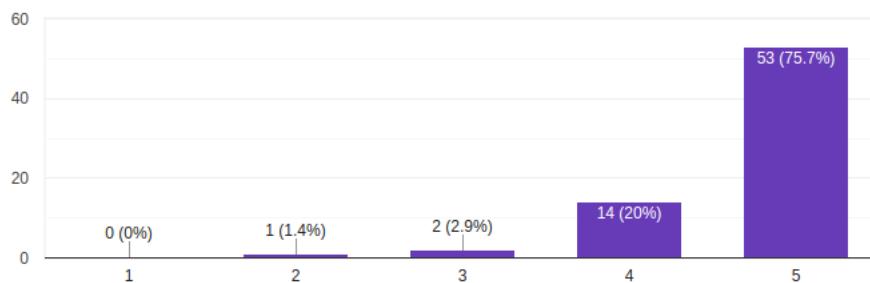


Gambar 5.10: Diagram penggunaan *email* di kalangan mahasiswa

Pada pertanyaan "Seberapa sering Anda membuka *email student* Anda?", mayoritas mahasiswa menjawab 5 (sering sekali). Jumlah mahasiswa yang menjawab 5 (sering sekali) adalah 24 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.10.

Seberapa sering Anda membuka aplikasi LINE?

70 responses

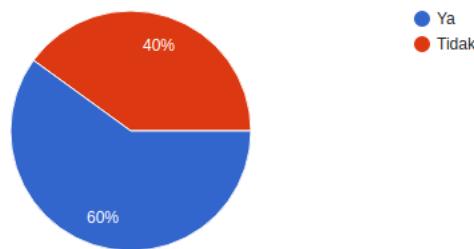


Gambar 5.11: Diagram penggunaan LINE di kalangan mahasiswa

Pada pertanyaan "Seberapa sering Anda membuka aplikasi LINE?", mayoritas mahasiswa menjawab 5 (sering sekali). Jumlah mahasiswa yang menjawab 5 (sering sekali) adalah 53 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.11.

Apakah Anda pernah menerima notifikasi pemberitahuan dari akun bot shadowtape pada jam 12 siang setelah mengikuti akun tersebut?

70 responses

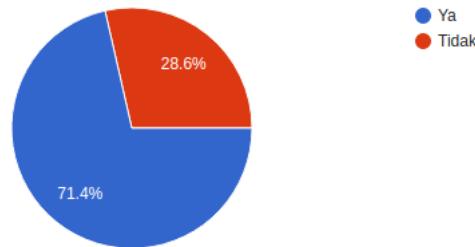


Gambar 5.12: Diagram mahasiswa yang menerima notifikasi LINE

Pada pertanyaan "Apakah Anda pernah menerima notifikasi pemberitahuan dari akun bot shadowtape pada jam 12 siang setelah mengikuti akun tersebut?", 42 mahasiswa menjawab "Ya" dan 28 mahasiswa menjawab "Tidak". Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.12. Mahasiswa yang menjawab "Tidak" belum tentu telah mengikuti *bot* karena jumlah maksimal teman akun LINE@ shadowtape adalah 50 akun. Peneliti memperkirakan jumlah mahasiswa yang tidak mendapatkan notifikasi LINE setelah mengikuti bot LINE tidak lebih dari 8 mahasiswa.

Apakah Anda dapat membuka url yang dicantumkan di notifikasi pengumuman?

42 responses



Gambar 5.13: Diagram mahasiswa yang dapat membuka URL pengumuman

Pada pertanyaan "Apakah Anda dapat membuka url yang dicantumkan di notifikasi pengumuman?", 30 mahasiswa menjawab "Ya" dan 12 mahasiswa menjawab "Tidak". Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.13. Peneliti menemukan beberapa mahasiswa yang menjawab "Tidak" adalah mahasiswa angkatan 2017 yang masih tidak dapat *login* setelah membaca kolom saran.

Jika Anda belum login saat membuka url yang dicantumkan di notifikasi pengumuman, apakah Anda diarahkan kembali ke url tersebut?

30 responses

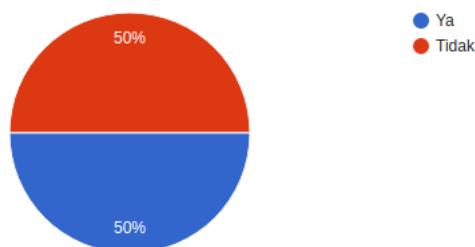


Gambar 5.14: Diagram mahasiswa yang diarahkan kembali ke URL pengumuman setelah *login*

Pada pertanyaan "Jika Anda belum login saat membuka url yang dicantumkan di notifikasi pengumuman, apakah Anda diarahkan kembali ke url tersebut?", 21 mahasiswa menjawab "Ya, saya diarahkan kembali ke url pengumuman yang bersangkutan." dan 9 mahasiswa menjawab "Tidak, saya diarahkan ke halaman utama atau halaman lain.". Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.14.

Apakah Anda dapat melihat pengumuman yang Anda kirimkan di menu pengumuman shadowtape
[\(<https://shadowtape.herokuapp.com/pengumuman>\)](https://shadowtape.herokuapp.com/pengumuman) pada jam 12 siang setelah Anda mengirimkan pengumuman?

2 responses

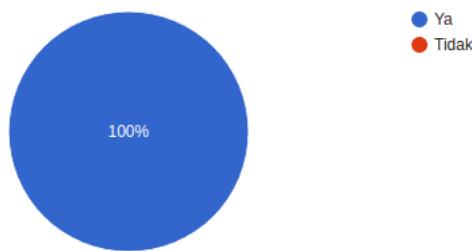


Gambar 5.15: Diagram dosen yang dapat melihat pengumuman yang diumumkannya

Pada pertanyaan "Apakah Anda dapat melihat pengumuman yang Anda kirimkan di menu pengumuman shadowtape (<https://shadowtape.herokuapp.com/pengumuman>) pada jam 12 siang setelah Anda mengirimkan pengumuman?", 1 dosen menjawab "Ya" dan 1 dosen menjawab "Tidak". Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.15. Dosen yang menjawab "Ya" adalah Pak Pascal Alfadian, sedangkan dosen yang menjawab "Tidak" adalah Pak Keenan Adiwijaya Leman. Pak Pascal pernah mengirimkan *email* ke shadowbluetape@gmail.com pada tanggal 24 April 2019, sedangkan Pak Keenan tidak pernah.

Apakah judul pengumuman sesuai dengan subjek email yang Anda kirimkan?

1 response

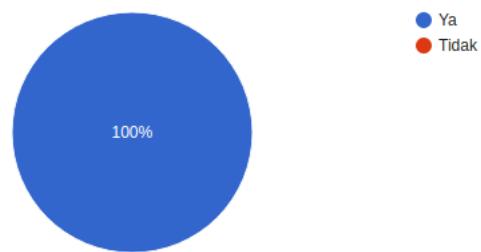


Gambar 5.16: Diagram dosen yang judul pengumumannya sesuai dengan subjek *email* yang ia kirim

Pada pertanyaan "Apakah judul pengumuman sesuai dengan subjek *email* yang Anda kirimkan?", Pak Pascal Alfadian menjawab "Ya". Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.16.

Apakah isi email sesuai dengan yang Anda kirimkan?

1 response

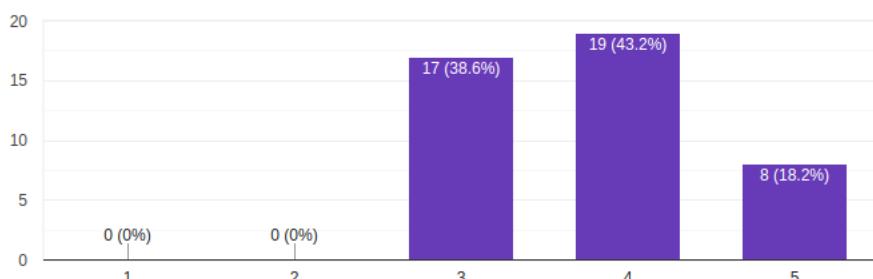


Gambar 5.17: Diagram dosen yang isi pengumumannya sesuai dengan isi *email* yang ia kirim

Pada pertanyaan "Apakah isi *email* sesuai dengan yang Anda kirimkan?", Pak Pascal Alfadian menjawab "Ya". Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.17.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya ingin menggunakan fitur ini" ?

44 responses

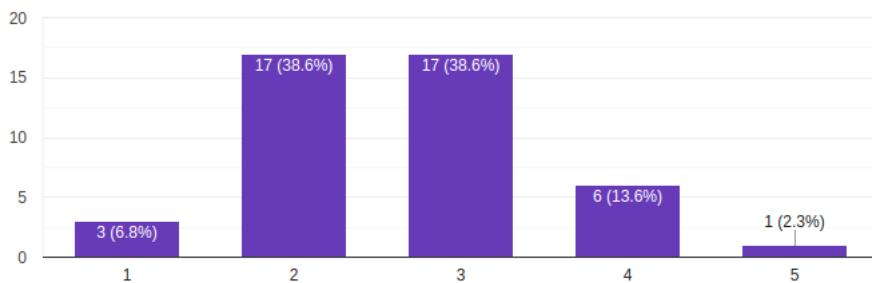


Gambar 5.18: Diagram jawaban pertanyaan *System Usability Scale* yang pertama

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya ingin menggunakan fitur ini" ?", mayoritas responden menjawab 4 (setuju). Jumlah responden yang menjawab 4 (setuju) adalah 19 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.18.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu rumit" ?

44 responses

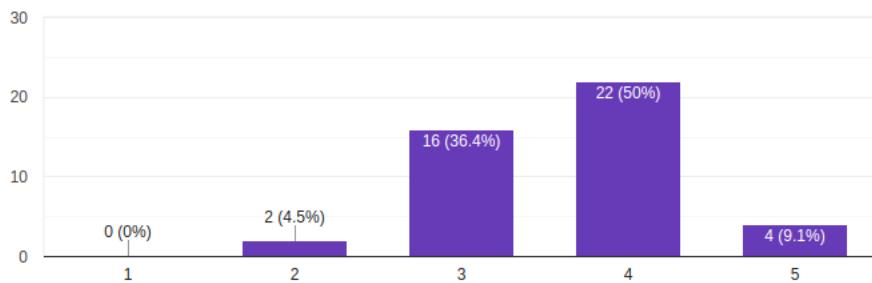


Gambar 5.19: Diagram jawaban pertanyaan *System Usability Scale* yang kedua

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu rumit" ?", mayoritas responden menjawab 2 (kurang setuju) dan 3(neutra). Jumlah responden yang menjawab 2 (kurang setuju) dan 3(neutra) masing-masing 17 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.19.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini mudah digunakan" ?

44 responses

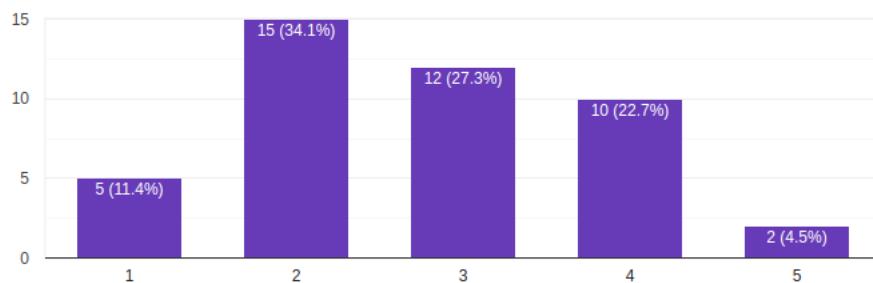


Gambar 5.20: Diagram jawaban pertanyaan *System Usability Scale* yang ketiga

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini mudah digunakan" ?", mayoritas responden menjawab 4 (setuju). Jumlah responden yang menjawab 4 (setuju) adalah 22 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.20.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya butuh bantuan teknisi untuk menggunakan fitur ini" ?

44 responses

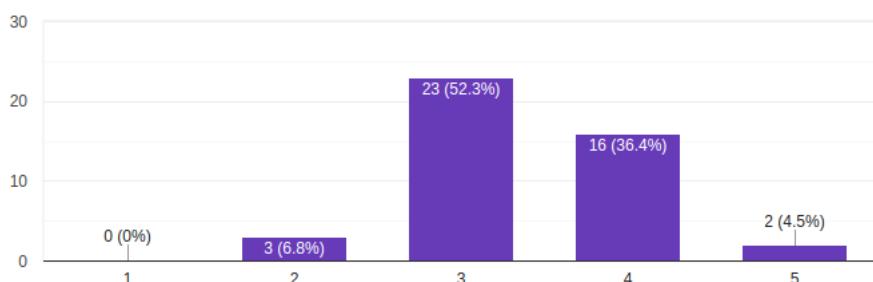


Gambar 5.21: Diagram jawaban pertanyaan *System Usability Scale* yang keempat

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya butuh bantuan teknisi untuk menggunakan fitur ini" ?", mayoritas responden menjawab 2 (kurang setuju). Jumlah responden yang menjawab 2 (kurang setuju) adalah 15 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.21.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir berbagai fungsi yang ada di fitur ini terintegrasi dengan baik" ?

44 responses

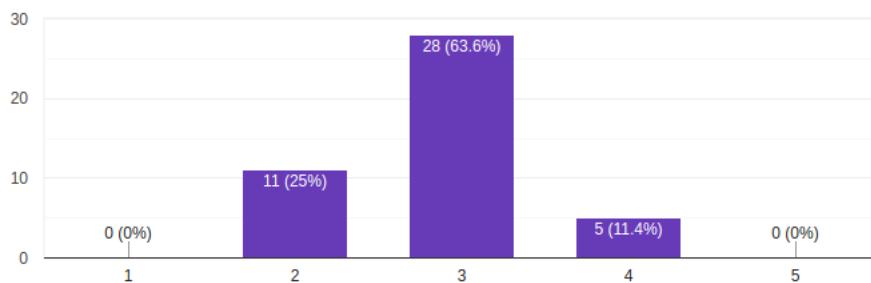


Gambar 5.22: Diagram jawaban pertanyaan *System Usability Scale* yang kelima

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir berbagai fungsi yang ada di fitur ini terintegrasi dengan baik" ?", mayoritas responden menjawab 3 (netral). Jumlah responden yang menjawab 3 (netral) adalah 23 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.22.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak hal yang TIDAK konsisten pada fitur ini" ?

44 responses

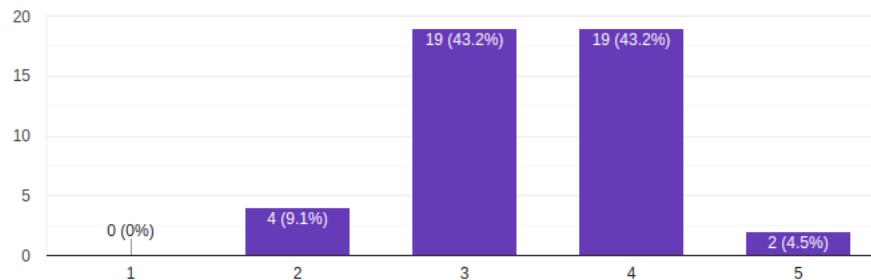


Gambar 5.23: Diagram jawaban pertanyaan *System Usability Scale* yang keenam

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak hal yang TIDAK konsisten pada fitur ini" ?", mayoritas responden menjawab 3 (netral). Jumlah responden yang menjawab 3 (netral) adalah 28 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.23.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak orang yang akan langsung mengerti cara memakai fitur ini" ?

44 responses

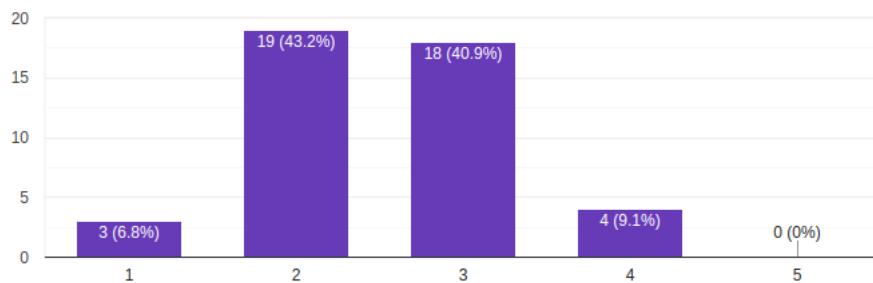


Gambar 5.24: Diagram jawaban pertanyaan *System Usability Scale* yang ketujuh

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak orang yang akan langsung mengerti cara memakai fitur ini" ?", mayoritas responden menjawab 3 (netral) dan 4 (setuju). Jumlah responden yang menjawab 3 (netral) dan 4 (setuju) masing-masing 19 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.24.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu sulit untuk digunakan" ?

44 responses

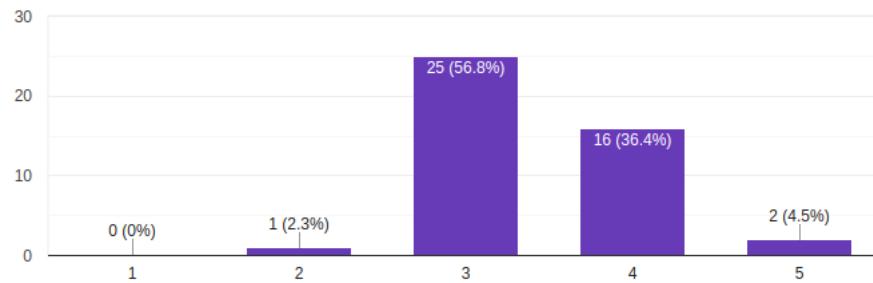


Gambar 5.25: Diagram jawaban pertanyaan *System Usability Scale* yang kedelapan

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu sulit untuk digunakan" ?", mayoritas responden menjawab 2 (kurang setuju). Jumlah responden yang menjawab 2 (kurang setuju) adalah 19 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.25.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini sangat nyaman digunakan" ?

44 responses

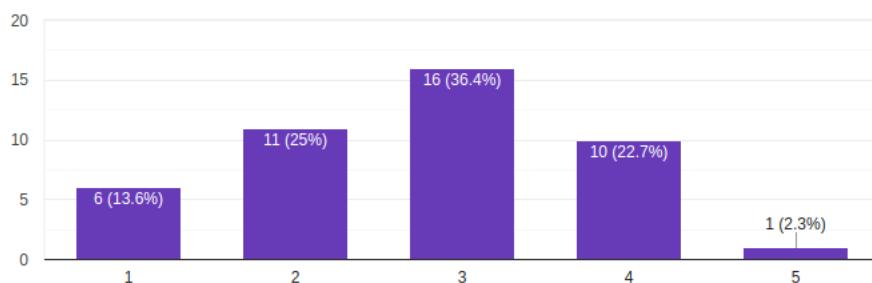


Gambar 5.26: Diagram jawaban pertanyaan *System Usability Scale* yang kesembilan

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini sangat nyaman digunakan" ?", mayoritas responden menjawab 3 (netral). Jumlah responden yang menjawab 3 (netral) adalah 25 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.26.

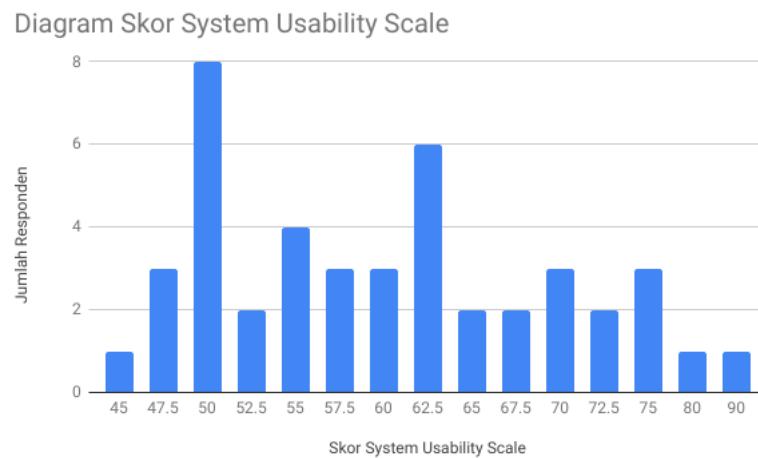
Apakah Anda setuju dengan pernyataan ini : "Saya perlu belajar banyak hal terlebih dahulu sebelum dapat menggunakan fitur ini" ?

44 responses



Gambar 5.27: Diagram jawaban pertanyaan *System Usability Scale* yang kesepuluh

Pada pertanyaan "Apakah Anda setuju dengan pernyataan ini : "Saya perlu belajar banyak hal terlebih dahulu sebelum dapat menggunakan fitur ini" ?", mayoritas responden menjawab 3 (netral). Jumlah responden yang menjawab 3 (netral) adalah 16 orang. Diagram jawaban untuk pertanyaan ini dapat dilihat pada Gambar 5.27.



Gambar 5.28: Diagram Skor System Usability Scale

Setelah menghitung skor *System Usability Scale* seluruh responden yang menjawab pertanyaan *System Usability Scale*, peneliti mendapatkan rata-ratanya adalah 60,34. Angka tersebut menunjukkan bahwa usabilitas fitur kolektor pengumuman termasuk kelompok rata-rata. Diagram skor *System Usability Scale* dapat dilihat pada Gambar 5.28.

Dari kolom saran, peneliti mendapatkan beberapa saran. Tabel 5.4 menampilkan saran yang didapatkan peneliti.

Tabel 5.4: Tabel saran

Alamat email	Saran
keenan.leman@unpar.ac.id	Tidak bisa "add friend" karena akun telah mencapai batas banyak teman.
7316027@student.unpar.ac.id	Lebih diperbagus tampilannya sehingga menarik user untuk menggunakannya
7314028@student.unpar.ac.id	Good Job!
2017730022@student.unpar.ac.id	terdapat beberapa error dalam login, dimana email saya (@student.unpar) tidak memiliki akses
2017730047@student.unpar.ac.id	memperbaiki url login yang dikirimkan di shadowtape
2017730011@student.unpar.ac.id	saran saya, usahakan tidak menggunakan aplikasi luar lain seperti LINE. dikarenakan tidak semua orang menggunakan line
2017730037@student.unpar.ac.id	Link yang dibuka langsung dari line tidak memiliki hak akses
2017730008@student.unpar.ac.id	Fitur ini bagus, memudahkan mahasiswa dalam menerima notifikasi email. Namun, waktunya sebaiknya tidak hanya pada jam 12 siang karena waktu tersebut adalah saat orang-orang makan siang yang mungkin saja sedang tidak membuka gadget/aplikasi Line pada khususnya.
2017730044@student.unpar.ac.id	Log in masih tidak dapat dilakukan, terdapat pernyataan bahwa "alamat email" tidak memiliki hak akses ke pengumuman. Contohnya email yang saya gunakan "2017730044@student.unpar.ac.id" tidak memiliki hak akses ke pengumuman"
7316025@student.unpar.ac.id	"Lebih user friendly "

Setelah masa pengujian eksperimental selesai, peneliti menemukan bahwa layanan LINE@ akan diberhentikan. Saat pengujian eksperimental selesai dilaksanakan, pengguna LINE tidak bisa membuat akun LINE@ baru. Sebagai ganti LINE@, LINE membuat layanan baru tapi serupa bernama LINE Official Account. Akun LINE@ yang sudah ada akan diubah menjadi akun LINE Official Account pada bulan Juni 2019.

Perbedaan kedua layanan tersebut terletak pada metode penjualan mereka. LINE@ meminta bayaran apabila pemilik akun LINE@ ingin menambah banyak akun LINE yang dapat menambahkannya sebagai teman, tapi banyak pesan yang dapat dikirim tidak dibatasi. LINE Official Account meminta bayaran apabila pemilik akun LINE Official Account ingin menambah pesan yang dapat ia kirim, tapi jumlah akun yang dapat menambahkannya sebagai teman tidak dibatasi.

BAB 6

KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berikut adalah kesimpulan yang dapat diambil dari penelitian ini :

- Konsep dan cara kerja LINE@ dan Heroku telah dipelajari.
- BlueTape dapat dimodifikasi agar dapat berjalan di Heroku. BlueTape versi skripsi ini diberi nama shadowtape dan dapat diakses di <https://shadowtape.herokuapp.com/>.
- BlueTape telah memiliki fitur kolektor pengumuman. Fitur ini dapat melakukan sinkronisasi kotak masuk untuk alamat *email shadowbluetape@gmail.com* setiap jam. *Email* yang diidentifikasi sebagai pengumuman dapat ditampilkan di <https://shadowtape.herokuapp.com/pengumuman>. BlueTape dapat memunculkan notifikasi pada akun LINE@ Shadowtape.
- Fitur kolektor pengumuman telah teruji.

6.2 Saran

Berikut adalah saran untuk pengembangan lebih lanjut :

- Mengadaptasi konfigurasi LINE@ pada BlueTape ke layanan LINE Official Account.
- Meneliti lebih lanjut masalah *login* yang ditemukan saat pengujian eksperimental.
- Meningkatkan usabilitas fitur ini agar calon pengguna nyaman menggunakaninya.

DAFTAR REFERENSI

- [1] Heroku (2018) Heroku dev center. <https://devcenter.heroku.com>. 15 November 2018.
- [2] Vixie, P. Cron manual page, 4th berkeley distribution. *The information from the crontab section (below and including the table) was taken (unedited, but with small additions) from the crontab manual pages. Type man, 1.*
- [3] Gmail (2018) Gmail api. <https://developers.google.com/gmail/api/>. 19 November 2018.
- [4] Group, P. (2008) Php:imap-manual. <http://php.net/manual/en/book imap.php>. 19 November 2018.
- [5] LINE (2018) Line developer. <https://developers.line.me>. 14 November 2018.
- [6] Albert, W. dan Tullis, T. (2013) *Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics* Interactive Technologies. Elsevier Science.
- [7] Etzkorn, L. (2017) *Introduction to Middleware: Web Services, Object Components, and Cloud Computing*. CRC Press, Amerika Serikat.

LAMPIRAN A

KODE PROGRAM

Kode program untuk fitur kolektor pengumuman diketik di dalam *file* yang sudah ada dan *file* baru. Isi dari beberapa *file* yang sudah ada tidak ditampilkan seluruhnya pada lampiran ini, melainkan ditampilkan sebagian dalam format *diff*. *File* dalam format *diff* adalah hasil perbandingan antara isi *file* sebelum diubah dengan isi *file* pada perubahan terakhir memakai *git diff*. Isi *file* yang hampir seluruh barisnya berubah tidak ditampilkan dalam format *diff*, melainkan ditampilkan seluruhnya. Keterangan *listing* untuk *file* dengan format *diff* diakhiri dengan kata ".diff".

Berikut *file* yang kode programnya diubah atau ditambah:

Listing A.1: .gitignore.diff

```
1| diff --git a/.gitignore b/.gitignore
2| index 14bc68c..38c8d7d 100644
3| --- a/.gitignore
4| +++ b/.gitignore
5| @@ -1 +1,2 @@
6| ./nbproject/private/
7| \ No newline at end of file
8| ./nbproject/private/
9| ./vendor
```

Listing A.2: Procfile

```
1| web: vendor/bin/heroku-php-apache2 www/
```

Listing A.3: composer.json.diff

```
1| diff --git a/composer.json b/composer.json
2| index 8bcb1fd..b55200b 100644
3| --- a/composer.json
4| +++ b/composer.json
5| @@ -1,6 +1,8 @@
6| {
7|     "require": {
8|         "google/apiclient": "^1.0",
9|         "phpoffice/phpexcel": "^1.8"
10|     }
11|     "ext-imap": "*",
12|     "phpoffice/phpexcel": "^1.8",
13|     "linecorp/line-bot-sdk": "^3.6"
14| }
```

Listing A.4: /www/application/config/auth.php

```
1| <?php
2|
3| defined('BASEPATH') OR exit('No direct script access allowed');
4|
5| $config['domain'] = getenv('CI_BASE_URL');
6| $config['google-clientid'] = getenv('GOOGLE_CLIENTID');
7| $config['google-clientsecret'] = getenv('GOOGLE_CLIENTSECRET');
8| $config['google-redirecturi'] = $config['domain'] . '/auth/oauth2callback';
9|
10| $config['email-config'] = Array(
11|     'protocol' => 'smtp',
12|     'smtp_host' => getenv('SMTP_HOST'),
13|     'smtp_port' => intval(getenv('SMTP_PORT')),
14|     'smtp_user' => getenv('SMTP_USER'),
15|     'smtp_pass' => getenv('SMTP_PASS'),
16|     'mailtype' => 'html',
17|     'charset' => 'iso-8859-1'
18| );
```

Listing A.5: /www/application/config/autoload.php.diff

```

1 diff --git a/www/application/config/autoload.php b/www/application/config/autoload.php
2 index 66cb5fa..5d8982b 100644
3 --- a/www/application/config/autoload.php
4 +++ b/www/application/config/autoload.php
5 @@ -58,7 +58,7 @@ $autoload['packages'] = array();
6 |
7 | $autoload['libraries'] = array('user_agent' => 'ua');
8 */
9 -$autoload['libraries'] = array('session');
10 +$autoload['libraries'] = array('database', 'session');
11 /*
12 | -----
13 |

```

Listing A.6: /www/application/config/config.php.diff

```

1 diff --git a/www/application/config/config.php b/www/application/config/config.php
2 index 07d2eee..cd6385c 100644
3 --- a/www/application/config/config.php
4 +++ b/www/application/config/config.php
5 @@ -23,7 +23,7 @@ defined('BASEPATH') OR exit('No_direct_script_access_allowed');
6 | a PHP script and you can easily do that on your own.
7 |
8 */
9 -$config['base_url'] = $_ENV['CI_BASE_URL'];
10 +$config['base_url'] = getenv('CI_BASE_URL');
11 /*
12 | -----
13 | -----
14 @@ -213,7 +213,7 @@ $config['directory_trigger'] = 'd';
15 | your log files will fill up very fast.
16 |
17 */
18 -$config['log_threshold'] = 0;
19 +$config['log_threshold'] = array(1, 3);
20 /*
21 | -----
22 | -----
23 @@ -439,7 +439,11 @@ $config['global_xss_filtering'] = FALSE;
24 | 'csrf_regenerate' - Regenerate token on every submission
25 | 'csrf_exclude_uris' - Array of URIs which ignore CSRF checks
26 */
27 -$config['csrf_protection'] = TRUE;
28 +if (stripos($_SERVER['REQUEST_URI'], '/PengumumanLine/webhook') === FALSE) {
29 + $config['csrf_protection'] = TRUE;
30 +else{
31 + $config['csrf_protection'] = FALSE;
32 +}
33 $config['csrf_token_name'] = 'csrf_token';
34 $config['csrf_cookie_name'] = 'csrf_token';
35 $config['csrf_expire'] = 7200;
36 @@ -512,3 +516,5 @@ $config['rewrite_short_tags'] = FALSE;
37 | Array: array('10.0.1.200', '192.168.5.0/24')
38 */
39 $config['proxy_ips'] = '';
40 +
41 +$config['sess_save_path'] = sys_get_temp_dir();

```

Listing A.7: /www/application/config/database.php

```

1 <?php
2 defined('BASEPATH') OR exit('No_direct_script_access_allowed');
3 /*
4 | -----
5 | DATABASE CONNECTIVITY SETTINGS
6 | -----
7 | This file will contain the settings needed to access your database.
8 |
9 | For complete instructions please consult the 'Database Connection'
10 | page of the User Guide.
11 |
12 |
13 | -----
14 | EXPLANATION OF VARIABLES
15 | -----
16 |
17 | ['dsn'] The full DSN string describe a connection to the database.
18 | ['hostname'] The hostname of your database server.
19 | ['username'] The username used to connect to the database
20 | ['password'] The password used to connect to the database
21 | ['database'] The name of the database you want to connect to
22 | ['dbdriver'] The database driver. e.g.: mysqli.
23 |     Currently supported:
24 |         cubrid, ibase, mssql, mysql, mysqli, oci8,
25 |         odbc, pdo, postgre, sqlite, sqlite3, sqlsrv
26 | ['dbprefix'] You can add an optional prefix, which will be added
27 |             to the table name when using the Query Builder class
28 | ['pconnect'] TRUE/FALSE - Whether to use a persistent connection
29 | ['db_debug'] TRUE/FALSE - Whether database errors should be displayed.
30 | ['cache_on'] TRUE/FALSE - Enables/disables query caching
31 | ['cachedir'] The path to the folder where cache files should be stored
32 | ['char_set'] The character set used in communicating with the database
33 | ['dbcollat'] The character collation used in communicating with the database
34 |             NOTE: For MySQL and MySQLi databases, this setting is only used
35 |                 as a backup if your server is running PHP < 5.2.3 or MySQL < 5.0.7

```

```

36 |     (and in table creation queries made with DB Forge).
37 |     There is an incompatibility in PHP with mysql_real_escape_string() which
38 |     can make your site vulnerable to SQL injection if you are using a
39 |     multi-byte character set and are running versions lower than these.
40 |     Sites using Latin-1 or UTF-8 database character set and collation are unaffected.
41 | ['swap_pre'] A default table prefix that should be swapped with the dbprefix
42 | ['encrypt'] Whether or not to use an encrypted connection.
43 |
44 |     'mysql' (deprecated), 'sqlsrv' and 'pdo/sqlsrv' drivers accept TRUE/FALSE
45 |     'mysqli' and 'pdo/mysql' drivers accept an array with the following options:
46 |
47 |         'ssl_key'      - Path to the private key file
48 |         'ssl_cert'     - Path to the public key certificate file
49 |         'ssl_ca'       - Path to the certificate authority file
50 |         'ssl_capath'   - Path to a directory containing trusted CA certificates in PEM format
51 |         'ssl_cipher'   - List of *allowed* ciphers to be used for the encryption, separated by colons (':')
52 |         'ssl_verify'   - TRUE/FALSE; Whether verify the server certificate or not ('mysqli' only)
53 |
54 | ['compress'] Whether or not to use client compression (MySQL only)
55 | ['stricton'] TRUE/FALSE - forces 'Strict Mode' connections
56 |                 - good for ensuring strict SQL while developing
57 | ['ssl_options'] Used to set various SSL options that can be used when making SSL connections.
58 | ['failover'] array - A array with 0 or more data for connections if the main should fail.
59 | ['save_queries'] TRUE/FALSE - Whether to "save" all executed queries.
60 |                 NOTE: Disabling this will also effectively disable both
61 | $this->db->last_query() and profiling of DB queries.
62 | When you run a query, with this setting set to TRUE (default),
63 | CodeIgniter will store the SQL statement for debugging purposes.
64 | However, this may cause high memory usage, especially if you run
65 | a lot of SQL queries ... disable this to avoid that problem.
66 |
67 | The $active_group variable lets you choose which connection group to
68 | make active. By default there is only one group (the 'default' group).
69 |
70 | The $query_builder variables lets you determine whether or not to load
71 | the query builder class.
72 */
73 $active_group = 'default';
74 $query_builder = TRUE;
75
76 $db['default'] = array(
77     'dsn'      => '',
78     'hostname' => getenv('CI_DB_HOSTNAME'),
79     'username' => getenv('CI_DB_USERNAME'),
80     'password' => getenv('CI_DB_PASSWORD'),
81     'database' => getenv('CI_DB_DATABASE'),
82     'dbdriver' => 'postgre',
83     'dbprefix' => '',
84     'pconnect' => FALSE,
85     'db_debug' => (ENVIRONMENT !== 'production'),
86     'cache_on' => FALSE,
87     'cachedir' => '',
88     'char_set' => 'utf8',
89     'dbcollat' => 'utf8_general_ci',
90     'swap_pre' => '',
91     'encrypt' => FALSE,
92     'compress' => FALSE,
93     'stricton' => FALSE,
94     'failover' => array(),
95     'save_queries' => TRUE
96 );

```

Listing A.8: /www/application/config/modules.php.diff

```

1| diff --git a/www/application/config/modules.php b/www/application/config/modules.php
2| index 5dfb8b1..db58232 100644
3| --- a/www/application/config/modules.php
4| +++ b/www/application/config/modules.php
5| @@ -8,8 +8,8 @@ $config['module-names'] = array(
6|     'PerubahanKuliahRequest' => 'Perubahan_Kuliah',
7|     'PerubahanKuliahManage' => 'Manajemen_Perubahan_Kuliah',
8|     'EntriJadwalDosen' => 'Entri_Jadwal_Dosen',
9|     'LihatJadwalDosen' => 'Lihat_Jadwal_Dosen'
10| -
11| +     'LihatJadwalDosen' => 'Lihat_Jadwal_Dosen',
12| +     'Pengumuman' => 'Pengumuman'
13| );
14|
15| $config['modules'] = array(
16| @@ -18,12 +18,13 @@ $config['modules'] = array(
17|     'PerubahanKuliahRequest' => array('root', 'staf.unpar'),
18|     'PerubahanKuliahManage' => array('root', 'tu.ftis'),
19|     'EntriJadwalDosen' => array('root', 'dosen.informatika' ),
20|     'LihatJadwalDosen' => array('root', 'mahasiswa.informatika', 'dosen.informatika'),
21|     'LihatJadwalDosen' => array('root', 'mahasiswa.informatika', 'dosen.informatika'),
22|     'Pengumuman' => array('root', 'mahasiswa.informatika', 'dosen.informatika')
23| );
24|
25| $config['roles'] = array(
26|     'root' => array('pascal@unpar.ac.id', 'shao.wei@unpar.ac.id'),
27|     'tu.ftis' => array('shao.wei@unpar.ac.id', 'pranyoto@unpar.ac.id', 'walip@unpar.ac.id', 'dwina@unpar.ac.id'),
28|     'root' => array('pascal@unpar.ac.id', 'shao.wei@unpar.ac.id', '7315029@student.unpar.ac.id'),
29|     'tu.ftis' => array('pascal@unpar.ac.id'), // array('shao.wei@unpar.ac.id', 'pranyoto@unpar.ac.id', 'walip@unpar.ac.id', 'dwina@unpar.ac.id'),
30|     'mahasiswa.ftis' => '[7[123]\\d{5}]|(20[1-9][0-9]7[123][0-9]{4})|(61[678][0-9]{7})@student\\.unpar\\.ac\\\\.id',
31|     'staf.unpar' => '.+@unpar\\.ac\\\\.id',
32|     'dosen.informatika' => array ('cheni@unpar.ac.id', 'mariskha@unpar.ac.id', 'anung@unpar.ac.id', 'moertini@unpar.ac.id', 'natalia@unpar.ac.id', 'chandraw@unpar.ac.id', 'elisatih@unpar.ac.id', 'gkarya@unpar.ac.id', 'husnulhakim@unpar.ac.id', '

```

```
joanna@unpar.ac.id', 'lionov@unpar.ac.id', 'luciana@unpar.ac.id', 'pascal@unpar.ac.id', 'rosad5@unpar.ac.id', 'vania.
natali@unpar.ac.id', 'kristopher.h@unpar.ac.id', 'raymond.chandra@unpar.ac.id', 'keenan.leman@unpar.ac.id'),
```

Listing A.9: /www/application/config/pengumuman.php

```
1 | <?php
2 |
3 | defined('BASEPATH') OR exit('No_direct_script_access_allowed');
4 |
5 | $config['pengirimTerverifikasi'] = array(
6 |     'shadowbluetape@gmail.com', 'cheni@unpar.ac.id', 'mariskha@unpar.ac.id', 'anung@unpar.ac.id', 'moertini@unpar.ac.id', '
7 |     natalia@unpar.ac.id', 'chandraw@unpar.ac.id', 'elisatih@unpar.ac.id', 'gkarya@unpar.ac.id', 'husnulhakim@unpar.ac.id', '
8 |     joanna@unpar.ac.id', 'lionov@unpar.ac.id', 'luciana@unpar.ac.id', 'claudio.franciscus@unpar.ac.id', 'pascal@unpar.ac.id'
9 |     , 'rosad5@unpar.ac.id', 'vania.natali@unpar.ac.id', 'kristopher.h@unpar.ac.id', 'raymond.chandra@unpar.ac.id', 'keenan.
10 |     leman@unpar.ac.id'
11 | );
12 | 
```

Listing A.10: /www/application/config/routes.php.diff

```
1 | diff --git a/www/application/config/routes.php b/www/application/config/routes.php
2 | index a19acda..8c388ea 100644
3 | --- a/www/application/config/routes.php
4 | +++ b/www/application/config/routes.php
5 | @@ -52,3 +52,5 @@ defined('BASEPATH') OR exit('No_direct_script_access_allowed');
6 |     $route['default_controller'] = 'auth';
7 |     $route['404_override'] = '';
8 |     $route['translate_uri_dashes'] = FALSE;
9 |
10 | +$route['pengumuman/page-(:num)'] = '/pengumuman/page/$1';
```

Listing A.11: /www/application/controllers/Auth.php.diff

```
1 | diff --git a/www/application/controllers/Auth.php b/www/application/controllers/Auth.php
2 | index da2c7b8..8c99c7d 100644
3 | --- a/www/application/controllers/Auth.php
4 | +++ b/www/application/controllers/Auth.php
5 | @@ -19,7 +19,12 @@ class Auth extends CI_Controller {
6 |     if ($code != NULL) {
7 |         $this->Auth_model->authenticateOAuthCode($code);
8 |         $userInfo = $this->Auth_model->getUserInfo();
9 |         header('Location:' . $userInfo['modules'][0]);
10 |         if ($this->session->has_userdata('redirect_url')){
11 |             redirect($this->session->userdata('redirect_url'));
12 |         }
13 |         else{
14 |             header('Location:' . $userInfo['modules'][0]);
15 |         }
16 |     } else {
17 |         throw new Exception("Mohon_login_terlebih_dahulu.");
18 |     }
```

Listing A.12: /www/application/controllers/Cron.php

```
1 | <?php
2 | defined('BASEPATH') OR exit('No_direct_script_access_allowed');
3 |
4 | class Cron extends CI_Controller {
5 |
6 |     public function daily() {
7 |         try {
8 |             $this->load->model('Pengumuman_model');
9 |             $newEmails = $this->Pengumuman_model->checkEmail();
10 |             $numberOfEmailsFound = 0;
11 |             $numberOfAnnouncementEmails = 0;
12 |             if ($newEmails != null){
13 |                 foreach ($newEmails as $newEmail){
14 |                     $numberOfEmailsFound = $numberOfEmailsFound + 1;
15 |                     $isPengumuman = $this->Pengumuman_model->proceedEmail($newEmail);
16 |                     if ($isPengumuman){
17 |                         $numberOfAnnouncementEmails = $numberOfAnnouncementEmails + 1;
18 |                     }
19 |                 }
20 |             }
21 |             log_message('info', "Successfully_performed_cron_jobs._The_number_of_new_emails_found_is_". $numberOfEmailsFound . "
22 |             and_the_number_of_new_emails_that_are_announcements_is_". $numberOfAnnouncementEmails . ".");
23 |             http_response_code(200);
24 |             echo json_encode([
25 |                 'message' => "Successfully_performed_cron_jobs."
26 |             ]);
27 |         } catch (Exception $e) {
28 |             log_message('error', $e->getMessage());
29 |             http_response_code(500);
30 |             echo json_encode([
31 |                 'message' => $e->getMessage()
32 |             ]);
33 |         }
34 |     }
35 | }
```

Listing A.13: /www/application/controllers/Pengumuman.php

```
1 | <?php
```

```

2 defined('BASEPATH') OR exit('No direct script access allowed');
3
4 class Pengumuman extends CI_Controller {
5
6     public function __construct() {
7         parent::__construct();
8         try {
9             $this->load->helper('url');
10            $this->session->set_userdata('redirect_url', current_url());
11            $this->Auth_model->checkModuleAllowed(get_class());
12        } catch (Exception $ex) {
13            $this->session->set_flashdata('error', $ex->getMessage());
14            header('Location:' . '/');
15        }
16        $this->load->library('BlueTape');
17        $this->load->model('Pengumuman_model');
18        $this->load->database();
19    }
20
21    public function index() {
22        // Retrieve logged in user data
23        $userInfo = $this->Auth_model->getUserInfo();
24
25        $this->db->select();
26        $this->db->order_by('id', 'desc');
27        $query = $this->db->get('Pengumuman');
28        $announcements = $query->result_array();
29        foreach ($announcements as $announcement) {
30            $announcement['url'] = "/pengumuman/read/" . $announcement['id'];
31        }
32
33        $this->page(1);
34    }
35
36    public function read($id){
37        $this->db->where('id', $id);
38        $this->db->select('*');
39        $this->db->from('Pengumuman');
40        $query = $this->db->get();
41        $pengumuman= $query->row_array();
42        if ($pengumuman === NULL) {
43            show_404();
44            exit;
45        }
46        $this->load->view('Pengumuman/read', array(
47            'currentModule' => get_class(),
48            'pengumuman' => $pengumuman
49        ));
50    }
51
52    public function page($page){
53        $limit = 10;
54        $this->pagination($page,$limit,((($page-1)*$limit));
55    }
56
57    public function pagination($page,$limit,$i){
58        $jumlahPengumuman = $this->db->count_all('Pengumuman');
59        $this->db->select('*');
60        $this->db->order_by('waktuTerkirim', 'desc');
61        $this->db->from('Pengumuman');
62        $this->db->limit($limit,$i);
63        $query = $this->db->get();
64        $pengumumans = $query->result_array();
65        $currentPage = $page;
66        $pengumumanPerPage = $limit;
67        $this->load->view('Pengumuman/main', array(
68            'currentModule' => get_class(),
69            'jumlahPengumuman' => $jumlahPengumuman,
70            'pengumumans' => $pengumumans,
71            'currentPage' => $currentPage,
72            'pengumumanPerPage' => $pengumumanPerPage
73        ));
74    }
75}

```

Listing A.14: /www/application/controllers/PengumumanLine.php

```

1 <?php
2 defined('BASEPATH') OR exit('No direct script access allowed');
3
4 class PengumumanLine extends CI_Controller {
5
6     public function webhook(){
7         try{
8             if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
9                 http_response_code(405);
10                error_log('Method not allowed');
11                exit();
12            }
13
14            $httpPostRequestBody = file_get_contents('php://input');
15
16            if (strlen($httpPostRequestBody) === 0) {
17                http_response_code(400);
18                error_log('Missing request body');
19                exit();
20            }
21

```

```

22     $xLineSignature = $_SERVER['HTTP_X_LINE_SIGNATURE'];
23
24     if (empty($xLineSignature)) {
25         http_response_code(400); // Bad Request, Signature is Missing
26     }
27     else{
28         $this->load->model('Pengumuman_Line_model');
29         $this->Pengumuman_Line_model->proceedWebhook($httpPostRequestBody, $xLineSignature);
30     }
31     http_response_code(200);
32 }
33 catch(Exception $e){
34     http_response_code(500);
35 }
36 }
37 }
```

Listing A.15: /www/application/core/MY_Log.php

```

1 <?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2 // this class is adapted from system/libraries/Log.php
3 /**
4 * CodeIgniter
5 *
6 * An open source application development framework for PHP 5.1.6 or newer
7 *
8 * @package      CodeIgniter
9 * @author       EllisLab Dev Team
10 * @copyright    Copyright (c) 2008 - 2014, EllisLab, Inc.
11 * @copyright    Copyright (c) 2014 - 2015, British Columbia Institute of Technology (http://bcit.ca/)
12 * @license      http://codeigniter.com/user_guide/license.html
13 * @link         http://codeigniter.com
14 * @since        Version 1.0
15 * @filesource
16 */
17
18 /**
19 * Logging Class
20 */
21 /**
22 * Logging Class
23 *
24 * @package      CodeIgniter
25 * @subpackage   Libraries
26 * @category    Logging
27 * @author       EllisLab Dev Team
28 * @link         http://codeigniter.com/user_guide/general/errors.html
29 */
30 class MY_Log extends CI_Log {
31
32     protected $_threshold    = 1;
33     protected $_date_fmt     = 'Y-m-d H:i:s';
34     protected $_levels       = array('ERROR' => '1', 'DEBUG' => '2', 'INFO' => '3', 'ALL' => '4');
35
36     /**
37     * Constructor
38     */
39     public function __construct()
40     {
41         $config =& get_config();
42
43         if (is_numeric($config['log_threshold']))
44         {
45             $this->_threshold = $config['log_threshold'];
46         }
47
48         if ($config['log_date_format'] != '')
49         {
50             $this->_date_fmt = $config['log_date_format'];
51         }
52     }
53
54 /**
55 * Write Log to php://stderr
56 *
57 * Generally this function will be called using the global log_message() function
58 *
59 * @param string the error level
60 * @param string the error message
61 * @param bool whether the error is a native PHP error
62 * @return bool
63 */
64 public function write_log($level = 'error', $msg, $php_error = FALSE)
65 {
66     $level = strtoupper($level);
67
68     if ( ! isset($this->_levels[$level]) OR ($this->_levels[$level] > $this->_threshold))
69     {
70         return FALSE;
71     }
72
73     file_put_contents('php://stderr', $level.'.'.($level == 'INFO' ? '-' : '.').$this->_date_fmt.'-->'.$msg."\n");
74
75     return TRUE;
76 }
77 }
```

```

79 }
80 // END Log Class
81
82 /* End of file MY_Log.php */
83 /* Location: ./application/core/MY_Log.php */
84

```

Listing A.16: /www/application/migrations/20181011103200_Pengumuman_Initial.php

```

1 <?php
2
3 defined('BASEPATH') OR exit('No direct script access allowed');
4
5 class Migration_Pengumuman_initial extends CI_Migration {
6
7     public function up() {
8         $fields = array(
9             'id' => array(
10                 'type' => 'int',
11                 'auto_increment' => TRUE
12             ),
13             'namaPengirim' => array(
14                 'type' => 'VARCHAR',
15                 'constraint' => '256'
16             ),
17             'emailPengirim' => array(
18                 'type' => 'VARCHAR',
19                 'constraint' => '256'
20             ),
21             'waktuTerkirim' => array(
22                 'type' => 'timestamp'
23             ),
24             'subjek' => array(
25                 'type' => 'VARCHAR',
26                 'constraint' => '256'
27             ),
28             'isi' => array(
29                 'type' => 'TEXT',
30                 'null' => TRUE
31             ),
32             'ketersediaanLampiran' => array(
33                 'type' => 'VARCHAR',
34                 'constraint' => '1'
35             )
36         );
37         $this->dbforge->add_field($fields);
38         $this->dbforge->add_key('id', TRUE);
39         $this->dbforge->create_table('Pengumuman');
40     }
41
42     public function down() {
43
44     }
45 }
46

```

Listing A.17: /www/application/migrations/20190210224400_PengumumanLineFollowers_initial.php

```

1 <?php
2
3 defined('BASEPATH') OR exit('No direct script access allowed');
4
5 class Migration_PengumumanLineFollowers_initial extends CI_Migration {
6
7     public function up() {
8         $fields = array(
9             'userId' => array(
10                 'type' => 'VARCHAR',
11                 'constraint' => '256'
12             )
13         );
14         $this->dbforge->add_field($fields);
15         $this->dbforge->add_key('userId', TRUE);
16         $this->dbforge->create_table('PengumumanLineFollowers');
17     }
18
19     public function down() {
20
21     }
22 }
23

```

Listing A.18: /www/application/models/Pengumuman_model.php

```

1 <?php
2 defined('BASEPATH') OR exit('No direct script access allowed');
3
4 class Pengumuman_model extends CI_Model {
5     public function __construct() {
6         parent::__construct();
7
8         $this->load->config('auth');
9         $this->load->config('modules');
10    }
11

```

```

12| public function checkEmail(){
13|     $newEmails = null;
14|
15|     $hostname = getEnv('HOSTNAME_INCOMING_EMAIL');
16|     $username = getEnv('ANNOUNCEMENT_EMAIL');
17|     $password = getEnv('ANNOUNCEMENT_PASSWORD');
18|
19|     $inbox = imap_open($hostname,$username,$password) or die('Cannot connect to Gmail: ' . imap_last_error());
20|
21|     $emails = imap_search($inbox,'UNSEEN');
22|
23|     if($emails) {
24|         $i = 0;
25|         foreach($emails as $emailNumber) {
26|             $invalid = false;
27|             $header = imap_headerinfo($inbox,$emailNumber);
28|             $from = isset($header->from) ? $header->from : null;
29|             if($from != null){
30|                 $fromaddress = null;
31|                 foreach($from as $id => $object){
32|                     $fromaddress = isset($object->mailbox) && isset($object->host) ? $object->mailbox . "@" . $object->host :
33|                                         null;
34|                 }
35|                 $bodymsg = '';
36|                 $attachmentExist = 'N';
37|                 $structure = imap_fetchstructure($inbox, $emailNumber);
38|                 if(isset($structure->parts) && is_array($structure->parts)) {
39|                     if(isset($structure->parts[1])){
40|                         $parts1 = $structure->parts[1];
41|                         if(isset($parts1->disposition) && $parts1->disposition == "ATTACHMENT"){
42|                             $attachmentExist = 'Y';
43|                             $parts0 = $structure->parts[0];
44|                             if(isset($parts0->parts[1])){
45|                                 $bodymsg = imap_qprint(imap_fetchbody($inbox, $emailNumber, '1.2'));
46|                             }
47|                             else if(isset($parts0->parts[0])){
48|                                 $bodymsg = imap_qprint(imap_fetchbody($inbox, $emailNumber, '1.1'));
49|                             }
50|                             else{
51|                                 $bodymsg = imap_qprint(imap_fetchbody($inbox, $emailNumber, '1'));
52|                             }
53|                             else{
54|                                 $bodymsg = imap_qprint(imap_fetchbody($inbox, $emailNumber, '2'));
55|                             }
56|                         }
57|                         else{
58|                             $bodymsg = imap_qprint(imap_fetchbody($inbox, $emailNumber, '1'));
59|                         }
60|                     }
61|                     else{
62|                         $bodymsg = imap_qprint(imap_fetchbody($inbox, $emailNumber, '1'));
63|                     }
64|
65|                     if($fromaddress != null){
66|                         $newEmails[$i]['emailFrom'] = $fromaddress;
67|                         $newEmails[$i]['from'] = isset($header->fromaddress) ? $header->fromaddress : $fromaddress;
68|                         if(isset($header->update)){
69|                             $newEmails[$i]['date'] = date("Y-m-d H:i:s", $header->update);
70|                         }
71|                         else{
72|                             $invalid = true;
73|                         }
74|
75|                         if(isset($header->subject)){
76|                             $newEmails[$i]['subject'] = $header->subject;
77|                         }
78|                         else{
79|                             $invalid = true;
80|                         }
81|
82|                         $newEmails[$i]['body'] = $bodymsg;
83|                         $newEmails[$i]['attachmentExist'] = $attachmentExist;
84|                     }
85|                     if($invalid){
86|                         unset($newEmails[$i]);
87|                     }
88|                     else{
89|                         $i++;
90|                     }
91|                 }
92|             }
93|         }
94|
95|         $errors = imap_errors();
96|
97|         imap_close($inbox);
98|
99|         return $newEmails;
100}
101
102 public function proceedEmail($newEmail){
103     $isPengumuman = false;
104     $this->config->load('pengumuman');
105     $terverifikasi = 0;
106     $daftarEmailTerverifikasi = $this->config->item('pengirimTerverifikasi');
107     foreach($daftarEmailTerverifikasi as $emailTerverifikasi){
108         if($newEmail['emailFrom'] == $emailTerverifikasi){
109             $terverifikasi = 1;

```

```
110    }
111}
112
113 if($terverifikasi == 1){
114     $isPengumuman = true;
115     $this->db->insert('Pengumuman', array(
116         'namaPengirim' => $newEmail['from'],
117         'emailPengirim' => $newEmail['emailFrom'],
118         'waktuTerkirim' => $newEmail['date'],
119         'subjek' => $newEmail['subject'],
120         'isi' => $newEmail['body'],
121         'ketersediaanLampiran' => $newEmail['attachmentExist']
122     ));
123     $justInserted = $this->db->select("*")->order_by('id','desc')->limit(1)->get('Pengumuman')->row();
124     $id = $justInserted->id;
125
126     $this->load->model('Pengumuman_Line_model');
127     $message = "Ada pengumuman baru dari " . $newEmail['from'] . ":" . $newEmail['subject'] . ". Silahkan klik link ini untuk melihatnya;" . base_url() . "pengumuman/read/" . $id;
128     $this->Pengumuman_Line_model->pushMessageToAllFollowers($message);
129 }
130
131 return $isPengumuman;
132 }
```

Listing A.19: /www/application/models/Pengumuman_Line_model.php

```

72         $this->db->insert('PengumumanLineFollowers', array(
73             'userId' => $event->getUserId()
74         ));
75         $this->bot->replyText($event->getReplyToken(), 'Terima_kasih_telah_mengikuti_akun_ini._Pengumuman_baru_di_
76             Bluetape_akan_diberitahukan_melalui_akun_ini.');
77     } elseif ($event instanceof JoinEvent) {
78         // Not handled
79     } elseif ($event instanceof LeaveEvent) {
80         // Not handled
81     } elseif ($event instanceof PostbackEvent) {
82         // Not handled
83     } elseif ($event instanceof BeaconDetectionEvent) {
84         // Not handled
85     } elseif ($event instanceof AccountLinkEvent) {
86         // Not handled
87     } elseif ($event instanceof UnknownEvent) {
88         http_response_code(400); // Invalid event type
89     } else {
90         http_response_code(400); // Invalid event type
91     }
92 } catch (InvalidSignatureException $e) {
93     http_response_code(400); // Invalid signature
94 } catch (InvalidEventRequestException $e) {
95     http_response_code(400); // Invalid event request
96 }
97 }
98
99 public function pushMessageToAllFollowers($text){
100     $tos = [];
101     $query = $this->db->get('PengumumanLineFollowers');
102     foreach ($query->result() as $row){
103         $tos[] = $row->userId;
104     }
105
106     $ref = new ReflectionClass('LINE\LINEBot\MessageBuilder\TextMessageBuilder');
107     $textMessageBuilder = $ref->newInstanceArgs(array_merge([$text]));
108     $this->bot->multicast($tos, $textMessageBuilder);
109 }
110 }
111 }
```

Listing A.20: /www/application/views/Pengumuman/main.php

```

1 <?php
2 defined('BASEPATH') OR exit('No_direct_script_access_allowed');
3 ?><!doctype html>
4 <html class="no-js" lang="en">
5     <?php $this->load->view('templates/head_loggedin'); ?>
6     <body>
7         <?php $this->load->view('templates/topbar_loggedin'); ?>
8         <?php $this->load->view('templates/flashmessage'); ?>
9         <?php $this->load->view('templates/script_foundation'); ?>
10        <div class="row">
11            <div class="medium-12_column">
12                <?php
13                    if($jumlahPengumuman == 0){
14                        echo "Tidak Ada Pengumuman";
15                    } else{
16                        foreach($pengumumans as $pengumuman):
17
18                            ><div class="callout">
19                                <h3><a href=<?="/pengumuman/read/" . $pengumuman['id']?> <?=$pengumuman['subjek']?></a></h3>
20                                <p>
21                                    by : <?=$pengumuman['namaPengirim']?> (<?=$pengumuman['emailPengirim']?>) on <?=$date("D, d M Y, H:i:s", strtotime($pengumuman['waktuTerkirim']))?>
22                                </p>
23                            </div>
24                        endforeach;
25                    if($currentPage>1){
26                        $previousPage = $currentPage-1;
27                        echo "<a href='/pengumuman/page-". $previousPage . "'>Previous</a>";
28                    }
29                    if($jumlahPengumuman > $currentPage*$pengumumanPerPage){
30                        $nextPage = $currentPage+1;
31                        echo "<a href='/pengumuman/page-". $nextPage . "'>Next</a>";
32                    }
33                }
34            </div>
35        </div>
36    </body>
37 </html>
38 
```

Listing A.21: /www/application/views/Pengumuman/read.php

```

1 <?php
2 defined('BASEPATH') OR exit('No_direct_script_access_allowed');
3 ?><!doctype html>
4 <html class="no-js" lang="en">
5     <?php $this->load->view('templates/head_loggedin'); ?>
6     <body>
7         <?php $this->load->view('templates/topbar_loggedin'); ?>
8         <?php $this->load->view('templates/flashmessage'); ?>
```

```
9  <?php $this->load->view('templates/script_foundation'); ?>
10 <div class="row">
11   <div class="medium-12_column">
12     <div class="callout">
13       <h3><?= $pengumuman['subjek']?></h3>
14       <p>
15         by : <?= $pengumuman['namaPengirim']?> (<?= $pengumuman['emailPengirim']?>) on <?=date("D,d_M_Y:H:i:s",
16           strtotime($pengumuman['waktuTerkirim']))?>
17         </p>
18         <br><br>
19         <?php
20           if(!strpos($pengumuman['isi'], "<div")){
21             <?= nl2br(nl2br($pengumuman['isi'])); ?>
22           }
23         <?php
24           else{
25             <?= $pengumuman['isi']; ?>
26           <?php
27             }
28           </p>
29           <br><br>
30           <?php if($pengumuman['ketersediaanLampiran'] == 'Y')?>
31             <p>
32               *) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung email student Anda untuk
33               mengunduhnya.
34             </p>
35           <?php
36             endif;
37           ?>
38         </div>
39       </div>
40     </div>
41   </body>
42 </html>
```


LAMPIRAN B

LAMPIRAN PENGUJIAN EKSPERIMENTAL

B.1 Kuesioner

Kuesioner diawali dengan pertanyaan :

Survey Skripsi Kolektor Pengumuman Informatika

Survei ini dibuat untuk melengkapi skripsi dengan topik "Kolektor Pengumuman Informatika". Topik ini bertujuan untuk membuat fitur yang dapat memberikan notifikasi melalui bot LINE apabila ada pengumuman baru di ruang lingkup jurusan Teknik Informatika. Topik ini dibangun sebagai fitur tambahan pada BlueTape (pada skripsi ini diuji dulu di shadowtape).

Your email address (7315029@student.unpar.ac.id) will be recorded when you submit this form.
Not you? [Switch account](#)

* Required

Apakah Anda dosen atau mahasiswa? *

- Dosen
 Mahasiswa

NEXT

Never submit passwords through Google Forms.

Gambar B.1: Kuesioner bagian pertama

Apabila responden memilih "Dosen", maka responden akan dialihkan ke bagian Dosen (Lampiran B.1.1). Apabila responden memilih "Mahasiswa", maka responden akan dialihkan ke bagian Mahasiswa (Lampiran B.1.2). Responden wajib mengisi semua pertanyaan di kuesioner ini kecuali pertanyaan saran.

B.1.1 Dosen

Apakah Anda dapat melihat pengumuman yang Anda kirimkan di menu pengumuman shadowtape (<https://shadowtape.herokuapp.com/pengumuman>) pada jam 12 siang setelah Anda mengirimkan pengumuman? *

- Ya
- Tidak

BACK **NEXT**

Never submit passwords through Google Forms.

Gambar B.2: Kuesioner bagian Dosen pertanyaan pertama

Apabila responden memilih jawaban "Ya" pada pertanyaan pertama (Gambar B.2), maka responden akan dialihkan ke pertanyaan kedua (Gambar B.3). Apabila responden memilih jawaban "Tidak", maka responden akan dialihkan ke bagian *System Usability Scale* dan Saran (Lampiran B.1.3) dan Saran (Gambar B.1.3).

Apakah judul pengumuman sesuai dengan subjek email yang Anda kirimkan? *

- Ya
- Tidak

Apakah isi email sesuai dengan yang Anda kirimkan? *

- Ya
- Tidak

BACK **NEXT**

Never submit passwords through Google Forms.

Gambar B.3: Kuesioner bagian Dosen pertanyaan kedua dan ketiga

Apabila responden memilih jawaban "Ya" pada pertanyaan ketiga (pertanyaan kedua di Gambar B.2), maka responden akan dialihkan ke pertanyaan keempat (Gambar B.4). Apabila responden memilih jawaban "Tidak", maka responden akan dialihkan ke bagian *System Usability Scale* dan Saran (Gambar B.1.3).

Bagian manakah yang tidak sesuai dengan isi email yang Anda kirimkan? *

- Tidak ada in line attachment (Contoh : gambar yang disisipkan di antara paragraf)
- Tidak ada attachment, namun terdapat keterangan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung email student Anda untuk mengunduhnya." di bawah isi email.
- Tidak ada attachment dan tidak ada keterangan "*) Pengumuman ini memiliki lampiran, silahkan memeriksa langsung email student Anda untuk mengunduhnya." di bawah isi email.
- Formatnya berbeda
- Isi email (dalam bentuk teks) tidak lengkap.
- Other: _____

[BACK](#)[NEXT](#)

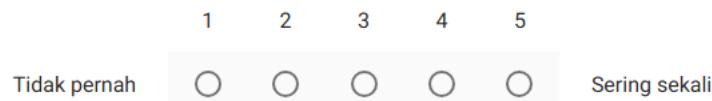
Never submit passwords through Google Forms.

Gambar B.4: Kuesioner bagian Dosen pertanyaan keempat

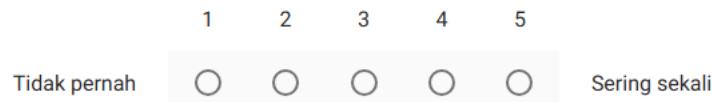
Apapun jawaban responden pada pertanyaan keempat (Gambar B.4), responden akan dialihkan ke bagian *System Usability Scale* dan Saran (Gambar B.1.3).

B.1.2 Mahasiswa

Seberapa sering Anda membuka email student Anda? *



Seberapa sering Anda membuka aplikasi LINE? *

[BACK](#)[NEXT](#)

Never submit passwords through Google Forms.

Gambar B.5: Kuesioner bagian Mahasiswa pertanyaan pertama dan kedua

Apapun jawaban responden pada pertanyaan pertama dan kedua(Gambar B.5), responden akan dialihkan ke pertanyaan kedua (Gambar B.6).

Apakah Anda pernah menerima notifikasi pemberitahuan dari akun bot shadowtape pada jam 12 siang setelah mengikuti akun tersebut? *

- Ya
- Tidak

[BACK](#) [NEXT](#)

Never submit passwords through Google Forms.

Gambar B.6: Kuesioner bagian Mahasiswa pertanyaan ketiga

Apabila responden memilih jawaban "Ya" pada pertanyaan ketiga (Gambar B.6), maka responden akan dialihkan ke pertanyaan keempat (Gambar B.7). Apabila responden memilih jawaban "Tidak", maka jawaban responden akan dikirim kepada peneliti.

Apakah Anda dapat membuka url yang dicantumkan di notifikasi pengumuman?

- Ya
- Tidak

[BACK](#) [NEXT](#)

Never submit passwords through Google Forms.

Gambar B.7: Kuesioner bagian Mahasiswa pertanyaan keempat

Apabila responden memilih jawaban "Ya" pada pertanyaan keempat (Gambar B.7), maka responden akan dialihkan ke pertanyaan kelima (Gambar B.8). Apabila responden memilih jawaban "Tidak", maka responden akan dialihkan ke bagian *System Usability Scale* dan Saran (Gambar B.1.3).

Jika Anda belum login saat membuka url yang dicantumkan di notifikasi pengumuman, apakah Anda diarahkan kembali ke url tersebut? *

- Ya, saya diarahkan kembali ke url pengumuman yang bersangkutan.
- Tidak, saya diarahkan ke halaman utama atau halaman lain.

[BACK](#) [NEXT](#)

Never submit passwords through Google Forms.

Gambar B.8: Kuesioner bagian Mahasiswa pertanyaan kelima

Apapun jawaban responden pada pertanyaan kelima (Gambar B.8), responden akan dialihkan

ke bagian *System Usability Scale* dan Saran (Gambar B.1.3).

B.1.3 *System Usability Scale* dan Saran

Bagian ini adalah bagian terakhir dari kuesioner. Bagian ini memiliki 11 pertanyaan yang terdiri dari 10 pertanyaan hasil adaptasi pertanyaan yang ditanyakan saat uji usabilitas dengan metode *System Usability Scale* dan 1 pertanyaan saran. Setelah responden menjawab bagian ini, jawaban responden akan dikirim kepada peneliti. Pertanyaan pada bagian ini ditampilkan pada Gambar B.9, Gambar B.10, dan Gambar B.11.

Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya ingin menggunakan fitur ini" ? *

1	2	3	4	5		
Sangat tidak setuju	<input type="radio"/>	Sangat setuju				

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu rumit" ? *

1	2	3	4	5		
Sangat tidak setuju	<input type="radio"/>	Sangat setuju				

Gambar B.9: Kuesioner bagian *System Usability Scale* dan Saran bagian 1

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini mudah digunakan" ? *

1 2 3 4 5

Sangat tidak setuju Sangat setuju

Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya butuh bantuan teknisi untuk menggunakan fitur ini" ? *

1 2 3 4 5

Sangat tidak setuju Sangat setuju

Apakah Anda setuju dengan pernyataan ini : "Saya pikir berbagai fungsi yang ada di fitur ini terintegrasi dengan baik" ? *

1 2 3 4 5

Sangat tidak setuju Sangat setuju

Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak hal yang TIDAK konsisten pada fitur ini" ? *

1 2 3 4 5

Sangat tidak setuju Sangat setuju

Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak orang yang akan langsung mengerti cara memakai fitur ini" ? *

1 2 3 4 5

Sangat tidak setuju Sangat setuju

Gambar B.10: Kuesioner bagian *System Usability Scale* dan Saran bagian 2

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu sulit untuk digunakan" ? *

1 2 3 4 5

Sangat tidak setuju

Sangat setuju

Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini sangat nyaman digunakan" ? *

1 2 3 4 5

Sangat tidak setuju

Sangat setuju

Apakah Anda setuju dengan pernyataan ini : "Saya perlu belajar banyak hal terlebih dahulu sebelum dapat menggunakan fitur ini" ? *

1 2 3 4 5

Sangat tidak setuju

Sangat setuju

Saran Anda untuk fitur ini

Your answer

BACK

NEXT

Never submit passwords through Google Forms.

Gambar B.11: Kuesioner bagian *System Usability Scale* dan Saran bagian 3

B.2 Hasil Mentah Kuesioner

Tabel B.1: Hasil Kuesioner Dosen

Kode Responden	(1)	(2)	(3)	(4)
b44980e1427379eb5135f9cf525caff7	Tidak	-	-	-
8759eb252d89ea97ae317b501d29e440	Ya	Ya	Ya	-

Keterangan:

- (1): Apakah Anda dapat melihat pengumuman yang Anda kirimkan di menu pengumuman shadowtape (<https://shadowtape.herokuapp.com/pengumuman>) pada jam 12 siang setelah Anda mengirimkan pengumuman?
- (2): Apakah judul pengumuman sesuai dengan subjek *email* yang Anda kirimkan?
- (3): Apakah isi *email* sesuai dengan yang Anda kirimkan?
- (4): Bagian manakah yang tidak sesuai dengan isi *email* yang Anda kirimkan?

Tabel B.2: Hasil Kuesioner Mahasiswa

Kode Responden	(1)	(2)	(3)	(4)	(5)
baa288c55dbe3374ae2024770ad7d71d	3	5	Ya	Tidak	-
30739dd65894243b2fde4fc2c60d7044	4	5	Tidak	-	-
da73fa4480f165dc734f00e9e751ed65	4	4	Tidak	-	-
6944991200f703e588f514c3132dfb00	3	5	Tidak	-	-
c303f0daa68202fce8598e871c3eb2e	3	5	Tidak	-	-
87cdf6dccc6c35b09177ee04d9aab8b	3	5	Tidak	-	-
890e85f2f19d33ae01904530aeee6820	5	5	Ya	Ya	Tidak
65ba791471cd92bf34cca127c3ea403a	4	5	Tidak	-	-
df46a7bc42efcc041f92e505359436ce	5	5	Tidak	-	-
5925f011062006b32ac89122779174ca	2	4	Tidak	-	-
bd86d293b094f35a23df03ac2827f075	2	4	Tidak	-	-
7d9892b660298864ec00bbef71558b31	4	5	Ya	Tidak	-
f5571c6dbcc9b28db2c83d478a78a9c9	5	5	Tidak	-	-
65b18974ffeb02daa4c006b397909684	5	3	Tidak	-	-
8ba81c09f3812f3501484c69f3da9acb	5	5	Tidak	-	-
70d55b0e588cddfcc3f99e9ba5dfbe6f	5	5	Ya	Ya	Ya
213f43ee1f0a692bc2ac08065d2ebe7c	5	5	Ya	Tidak	-
77385bbf5895bebd0b968eb9b4ed5c3f	5	5	Tidak	-	-
489d475466182fc84cd9dd3f41b164e4	5	5	Tidak	-	-
52b6e8f11c3e3eb163e275b36105a7d5	3	5	Tidak	-	-
0990276356eea41db102ee71af597517	5	5	Ya	Ya	Tidak
67ae5929a29cf2426c2029cab8588f43	4	5	Tidak	-	-
f0d182a65849bfa7d75fc8647b936d16	3	4	Ya	Ya	Ya
2de1ac036956e987584dc9e5b6f0c1f3	4	5	Ya	Ya	Ya
c2dd4f611c682de1403a5e76a5cfbc36	5	5	Tidak	-	-
0515aec76d9cb24bb71779fd39a5536d	2	5	Ya	Ya	Tidak
70047b9642824e1d4d4135f441558096	3	4	Ya	Ya	Tidak
126f8264062ddefe6cae7006186060ee	4	5	Ya	Ya	Ya
80bdb4d1c0843dd858594a27329bb593	3	2	Ya	Ya	Tidak
2f0c1a3e29b01bd37e2b309dea70f3fb	3	5	Ya	Tidak	-
0f13c207bf7d400a8279d4171e9f7323	4	4	Ya	Ya	Ya
8c6eea6a3d6a22229d7128d5c8dbb3f0	4	4	Ya	Ya	Tidak
3551530a405d71432874a7d7ec20b487	3	4	Ya	Ya	Ya
22e61646b0c84b0adbb64d8cd702601a	5	5	Ya	Ya	Ya
0ca428e8512d5abc28691ee6b6fab9c0	3	4	Tidak	-	-
c8d6743add10aa3cc5daa5891309c308	4	4	Ya	Tidak	-
da162a78d7eb43e1f459e9be9b2eade9	3	4	Tidak	-	-
0e388d3dfcfbc4df0a1bae5f4bba21b1	5	5	Tidak	-	-
0ae60f92cbd7025cebd7c654f5ecfdd8	2	5	Tidak	-	-
54aa340d9bf02b2bcb2d3730651685d6	4	5	Tidak	-	-
e63be302cfdda4240ab92074cae097bb	4	5	Tidak	-	-
f30b7a22a6a4a608cef733a3487397e6	3	4	Tidak	-	-
67e4b050d7bebec24964ee6463720b20	4	5	Ya	Tidak	-
7a74ad5b48a16725d3855143a5a9e1e2	5	5	Ya	Ya	Ya
3dc80df57a6789d804cc9f5bdd4571af	3	5	Tidak	-	-
84e410ec5bc4fc5f4df3f6288afac1da	4	5	Ya	Ya	Ya
d9bd7e6c75000d796acbbfbe35c92119	5	5	Ya	Ya	Tidak

8b4214ca9002814916a13e9d2402c0cd	2	5	Ya	Ya	Tidak
10a3588fa7742eb0164052ee8eccb4f6	3	5	Tidak	-	-
b4dd92d894607dcc467b214ab4ab5126	4	5	Ya	Ya	Ya
2932f45e458fd3cfaf1faaa6ba8f4de19	5	5	Ya	Tidak	-
9e3e261421008daf528b610cbc8f3d8f	5	5	Ya	Ya	Ya
72d4476b25275bc0a17598aa972fc35e	5	5	Tidak	-	-
d1acef8604a478b8cfed30d68a59adf4	5	5	Ya	Ya	Ya
9f542d3a9072f82558b818f44fd6e87b	4	5	Ya	Tidak	-
35e573adc70393e6fb75b13682a5d021	5	5	Ya	Tidak	-
945aa389b10884f4a49d375dbdce6ec3	3	5	Ya	Ya	Ya
8676ecaab998b8e261e76194bc8cf720	5	5	Ya	Ya	Ya
887a27864b02f85b52edc003db7c13e5	4	5	Ya	Tidak	-
66ccabd866b2bebfb9b827ccbc899cc7a	5	5	Ya	Tidak	-
ab786d0f73ddcdc9b67971d27f325acc	3	3	Ya	Ya	Ya
0b62edc4d0fda04013b222d57fb6ef7e	5	5	Ya	Ya	Ya
fa0a7c8ddc207cd6623c62940aed0a13	5	5	Ya	Tidak	-
bfee6e9dd49be37984ee5f77645c4b7c	3	5	Ya	Ya	Tidak
d790220a48a31bb1571bf8878545b27e	4	4	Ya	Ya	Ya
2fcfed5b287fa43ea835aaad9eed681b0	4	5	Ya	Ya	Ya
ceb5b37582d5d44719d41adb8d7302be	4	5	Ya	Ya	Ya
b9ce36d7dd6281de4b83fd555d9f10e1	4	5	Ya	Ya	Ya
7c4fc052aa4c422a607e75818a920b39	4	4	Tidak	-	-
8d0c412f1931db48fc80c6305f4561c3	4	5	Ya	Ya	Ya

Keterangan:

- (1): Seberapa sering Anda membuka *email student* Anda?
- (2): Seberapa sering Anda membuka aplikasi LINE?
- (3): Apakah Anda pernah menerima notifikasi pemberitahuan dari akun bot shadowtape pada jam 12 siang setelah mengikuti akun tersebut?
- (4): Apakah Anda dapat membuka *url* yang dicantumkan di notifikasi pengumuman?
- (5): Jika Anda belum *login* saat membuka *url* yang dicantumkan di notifikasi pengumuman, apakah Anda diarahkan kembali ke *url* tersebut?

Tabel B.3: Hasil Kuesioner System Usability Scale (SUS)

Kode Responden	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
baa288c55dbe3374ae2024770ad7d71d	4	2	4	4	3	2	4	1	4	2
b44980e1427379eb5135f9cf525caff7	5	1	5	1	3	3	5	1	5	1
890e85f2f19d33ae01904530aeee6820	4	3	4	2	4	3	4	2	4	2
7d9892b660298864ec00bbef71558b31	3	3	3	4	3	3	3	3	3	3
70d55b0e588cddfcc3f99e9ba5dfbe6f	4	2	4	4	4	3	3	2	4	4
213f43ee1f0a692bc2ac08065d2ebe7c	4	2	4	3	3	3	3	3	3	3
0990276356eea41db102ee71af597517	4	4	4	3	4	3	5	2	3	2
f0d182a65849bfa7d75fc8647b936d16	3	2	2	1	4	3	3	3	3	3
2de1ac036956e987584dc9e5b6f0c1f3	4	3	4	4	4	3	4	4	3	4
0515aec76d9cb24bb71779fd39a5536d	3	4	3	1	3	3	4	2	3	3

70047b9642824e1d4d4135f441558096	3	4	3	3	2	3	3	3	3	3
126f8264062ddefe6cae7006186060ee	4	4	4	4	4	4	4	4	4	4
80bdb4d1c0843dd858594a27329bb593	3	3	3	3	3	3	3	3	3	4
2f0c1a3e29b01bd37e2b309dea70f3fb	4	2	4	2	4	3	4	2	4	1
0f13c207bf7d400a8279d4171e9f7323	4	2	4	2	4	2	4	2	4	2
8c6eea6a3d6a22229d7128d5c8dbb3f0	3	4	4	4	4	4	4	2	4	4
3551530a405d71432874a7d7ec20b487	3	2	4	3	4	3	3	2	3	2
22e61646b0c84b0adbb64d8cd702601a	4	1	3	3	3	2	4	3	3	3
c8d6743add10aa3cc5daa5891309c308	3	2	4	3	3	3	2	3	2	3
67e4b050d7beb24964ee6463720b20	3	3	4	2	3	3	3	3	3	3
7a74ad5b48a16725d3855143a5a9e1e2	5	5	5	5	5	4	4	4	4	5
84e410ec5bc4fc5f4df3f6288afac1da	5	1	5	3	3	3	4	2	4	2
d9bd7e6c75000d796acbbfbe35c92119	5	2	2	5	5	3	3	3	3	3
8b4214ca9002814916a13e9d2402c0cd	4	2	4	2	3	3	3	3	4	3
b4dd92d894607dcc467b214ab4ab5126	3	3	3	3	3	3	3	3	3	3
2932f45e458fd3cfa1faaa6ba8f4de19	4	4	3	4	2	3	3	3	3	2
9e3e261421008daf528b610cbc8f3d8f	4	2	3	3	3	2	4	2	4	4
d1acef8604a478b8cfed30d68a59adf4	3	3	3	3	3	3	3	3	3	3
9f542d3a9072f82558b818f44fd6e87b	4	2	4	2	3	3	3	3	3	2
35e573adc70393e6fb75b13682a5d021	4	3	3	2	3	3	4	2	3	3
945aa389b10884f4a49d375dbdce6ec3	3	3	3	2	3	4	4	4	4	4
8676ecaab998b8e261e76194bc8cf720	3	2	4	1	4	2	3	2	3	1
887a27864b02f85b52edc003db7c13e5	5	2	4	4	4	2	4	2	4	2
66ccabd866b2bebf9b827ccbc899cc7a	3	3	4	2	3	4	2	2	3	2
ab786d0f73ddcdc9b67971d27f325acc	5	3	4	1	4	3	4	1	4	1
0b62edc4d0fda04013b222d57fb6ef7e	3	3	3	3	3	3	3	3	3	3
fa0a7c8ddc207cd6623c62940aed0a13	5	2	4	4	4	3	4	2	3	4
bfee6e9dd49be37984ee5f77645c4b7c	4	3	3	2	2	2	2	2	3	1
d790220a48a31bb1571bf8878545b27e	4	2	5	2	3	2	2	2	4	3
2fcfd5b287fa43ea835aaad9eed681b0	3	2	3	2	3	2	3	2	3	1
ceb5b37582d5d44719d41adb8d7302be	4	3	3	4	3	3	3	3	3	3
b9ce36d7dd6281de4b83fd555d9f10e1	3	3	3	2	4	2	3	3	3	4
8d0c412f1931db48fc80c6305f4561c3	5	3	4	2	4	3	4	3	5	4
8759eb252d89ea97ae317b501d29e440	4	3	4	2	3	2	4	2	4	2

Keterangan:

- (1) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya ingin menggunakan fitur ini" ?
- (2) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu rumit" ?
- (3) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini mudah digunakan" ?
- (4) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir saya butuh bantuan teknisi untuk menggunakan fitur ini" ?
- (5) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir berbagai fungsi yang ada di fitur ini terintegrasi dengan baik" ?
- (6) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak hal yang TIDAK konsisten pada fitur ini" ?

- (7) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir banyak orang yang akan langsung mengerti cara memakai fitur ini" ?
- (8) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini terlalu sulit untuk digunakan" ?
- (9) = Apakah Anda setuju dengan pernyataan ini : "Saya pikir fitur ini sangat nyaman digunakan" ?
- (10) = Apakah Anda setuju dengan pernyataan ini : "Saya perlu belajar banyak hal terlebih dahulu sebelum dapat menggunakan fitur ini" ?

Tabel B.4: Hasil Kuesioner Saran

Kode Responden	Saran Anda untuk fitur ini
b44980e1427379eb5135f9cf525caff7	Tidak bisa " <i>add friend</i> " karena akun telah mencapai batas banyak teman.
70d55b0e588cddfcc3f99e9ba5dfbe6f	Lebih diperbagus tampilannya sehingga menarik <i>user</i> untuk menggunakannya
0f13c207bf7d400a8279d4171e9f7323	<i>Good Job!</i>
2932f45e458fd3cfa1faaa6ba8f4de19	terdapat beberapa <i>error</i> dalam <i>login</i> , dimana <i>email</i> saya (@student.unpar) tidak memiliki akses
d1acef8604a478b8cfed30d68a59adf4	memperbaiki <i>url login</i> yang dikirimkan di shadowtape
66ccabd866b2bebf9b827ccbc899cc7a	saran saya, usahakan tidak menggunakan aplikasi luar lain seperti LINE. dikarenakan tidak semua orang menggunakan line
0b62edc4d0fd04013b222d57fb6ef7e	<i>Link</i> yang dibuka langsung dari line tidak memiliki hak akses
fa0a7c8ddc207cd6623c62940aed0a13	Fitur ini bagus, memudahkan mahasiswa dalam menerima notifikasi <i>email</i> . Namun, waktunya sebaiknya tidak hanya pada jam 12 siang karena waktu tersebut adalah saat orang-orang makan siang yang mungkin saja sedang tidak membuka <i>gadget/aplikasi Line</i> pada khususnya.
d790220a48a31bb1571bf8878545b27e	<i>Log in</i> masih tidak dapat dilakukan, terdapat pernyataan bahwa "alamat <i>email</i> " tidak memiliki hak akses ke pengumuman. Contohnya <i>email</i> yang saya gunakan "2017730044@student.unpar.ac.id" tidak memiliki hak akses ke pengumuman"
2fced5b287fa43ea835aaad9eed681b0	"Lebih <i>user friendly</i> "