SKRIPSI

APLIKASI PENGINTEGRASI OUTLOOK CALENDAR DAN SLACK

Sandy Giovanni Sutiansen

NPM: 2015730041

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

UNDERGRADUATE THESIS

OUTLOOK CALENDAR AND SLACK INTEGRATION APP

Sandy Giovanni Sutiansen

NPM: 2015730041

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

LEMBAR PENGESAHAN

APLIKASI PENGINTEGRASI OUTLOOK CALENDAR DAN SLACK

Sandy Giovanni Sutiansen

NPM: 2015730041

Bandung, 13 Desember 2019

Menyetujui,

Pembimbing

Pascal Alfadian, M.Comp.

Ketua Tim Penguji	Anggota Tim Penguji
Luciana Abednego, M.T.	**Raymond Chandra, M.T.**

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

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

APLIKASI PENGINTEGRASI OUTLOOK CALENDAR DAN SLACK

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 13 Desember 2019

Meterai
Rp. 6000

Sandy Giovanni Sutiansen
NPM: 2015730041

ABSTRAK

Outlook Calendar adalah sebuah aplikasi yang berfungsi selayaknya kalender yang bisa diisi sebuah event oleh pengguna dari aplikasi tersebut. *Slack* sendiri adalah sebuah aplikasi yang berbasis *cloud* yang diperuntukkan sebagai wadah kolaborasi antar tim. Para pengguna bisa mengatur status yang akan dipasang di dalam akunnya. Namun terkadang pengguna lupa untuk mengubah statusnya agar tidak terganggu dengan pengguna lain yang berusaha mengirimkan pesan disaat pengguna itu sedang dalam keadaan genting atau di dalam suatu pertemuan.

Dengan permasalahan itu, maka skripsi ini membahas mengenai sebuah perangkat lunak yang mengintegrasikan antara *event-event* yang tercatat di dalam *Outlook Calendar* dengan status dalam akun pengguna dalam *Slack*. Perangkat lunak yang dibangun di dalam skripsi ini adalah perangkat lunak yang mengambil data event, dan ketika sedang ada *event* yang berjalan pada saat itu, maka status di dalam akun pengguna di *Slack* akan terganti agar meminimalisir pengguna lain berusaha untuk mengirimkan pesan kepada pengguna yang akan mengganggu dengan status yang sudah terpasang.

Perangkat lunak ini akan mengambil data *event* yang tersimpan di dalam *Outlook Calendar*, lalu akan diperiksa dengan waktu sekarang. Jika waktu *event* sedang berjalan sekarang, maka perangkat lunak akan mengubah status yang terdapat pada aplikasi *Slack* menjadi "*In a meeting*" dan status akan kembali lagi kosong jika *event* sudah selesai berjalan.

Pengujian pada skripsi ini dilakukan melalui 2 lingkungan yaitu perangkat lunak diuji coba di *workspace* tempat perangkat lunak ini didaftarkan pada aplikasi *Slack* dan diuji coba di *workspace* lain. Hasil pengujian menunjukkan fitur yang diharapkan telah berjalan dengan sesuai harapan, baik itu di dalam *workspace* tempat perangkat lunak didaftarkan, maupun di *workspace* lain. Perangkat lunak ini dapat mengubah status pada aplikasi *Slack* di *workspace* manapun oleh pengguna siapapun dengan catatan setiap pengguna hanya bisa mendaftarkan 1 akun *Slack* di 1 *workspace*.

Kata-kata kunci: *Outlook Calendar*, *Slack*, *Event*, Status

ABSTRACT

Outlook Calendar is an application that works like a calendar that can be filled by an event by the user of the application. Slack itself is a cloud-based application that is intended as a forum for collaboration between teams. Users can set the status that will be installed in their account. But sometimes the user forgets to change their status so that they don't interfere with other users who try to send messages when the user is in a critical situation or in a meeting.

With that problem, this thesis will discuss about a software that integrates the events recorded in Outlook Calendar with the status in the user's account in Slack. The software that will be built in this thesis is software that retrieves event data, and when an event is running at that time, the status in the user's account in Slack will be changed to minimize other users trying to send messages to users who will interfere with the status that is already installed.

This software will retrieve event data stored in Outlook Calendar, then it will be checked with the current time. If the time the event is running now, the software will change the status contained in the Slack application to " In a meeting " and the status will return to empty if the event has finished.

Testing in this thesis is done through 2 environments namely software tested at workspaces where this software is registered in the Slack application and tested on other workspaces. The test results show that features are expected to run as expected, both in the workspace where the software is registered, and in other workspaces. This software can change the status of the Slack application in any workspace by any user as long as each user can only register 1 slack account in 1 workspace.

Keywords: Outlook Calendar, Slack, Event, Status

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

KATA PENGANTAR

Puji Syukur penulis panjatkan kepada Tuhan Yesus karena atas berkat-Nya lah skripsi dengan judul "Aplikasi Pengintegrasi Outlook Calendar dan Slack" ini bisa selesai dengan baik dan tepat waktu. Penulis berharap skripsi dan perangkat lunak yang dibangun dapat berguna bagi orang yang membutuhkan dan juga dapat membantu bagi orang yang akan melanjutkan penelitian ini untuk selanjutnya. Tidak lupa juga penulis mengucapkan terima kasih kepada:

- Keluarga, khususnya orang tua dan juga kepada adik yang telah menyemangati penulis secara langsung maupun tidak langsung, serta membuat suasana senyaman mungkin.
- Pak Pascal Alfadian yang telah membantu membimbing penulis dalam proses pembuatan skripsi beserta perangkat lunaknya.
- Teman yang selalu mendukung: Andi Tangguh Kippi Nusantara, Reyner Alexander, Adriswara Dwikarkasa, Vincent Joel Sinatra, Yonathan Kristian, Yosua, Raymond Nagawijaya, Arlin Sasqia Puspa Shiffa.
- Admin Lab Ftis yang bersedia ikutserta dalam melakukan pengujian perangkat lunak ini.
- Rekan-rekan kerja di Talent Media yang menyemangati dan juga ikutserta dalam melakukan pengujian perangkat lunak.
- Pihak-pihak lain yang tidak bisa disebutkan satu persatu.

Bandung, Desember 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 . . . 2
- 1.3 Tujuan . . . 2
- 1.4 Batasan Masalah . . . 3
- 1.5 Metodologi . . . 3
- 1.6 Sistematika Pembahasan . . . 3

2 LANDASAN TEORI 5
- 2.1 *Microsoft Graph API* . . . 5
  - 2.1.1 User resource type . . . 6
  - 2.1.2 Event resource type . . . 14
  - 2.1.3 Lain-lain . . . 18
- 2.2 *Slack API* . . . 19
- 2.3 *Node.js* . . . 22
  - 2.3.1 *HTTP* . . . 22
  - 2.3.2 Express.js . . . 24
  - 2.3.3 Handlebars . . . 25
  - 2.3.4 Simple OAuth2 . . . 25
  - 2.3.5 jsonwebtoken . . . 27
  - 2.3.6 Isomorphic Fetch . . . 28
  - 2.3.7 Microsoft Graph Client Library . . . 28
  - 2.3.8 Slack Web API . . . 28
- 2.4 *Heroku* . . . 29
  - 2.4.1 Dyno . . . 29
  - 2.4.2 Heroku Scheduler . . . 30
  - 2.4.3 *Heroku Postgres* . . . 30

3 ANALISIS 33
- 3.1 Analisis Perangkat Lunak . . . 33
  - 3.1.1 Analisis Diagram Alir Sistem . . . 33
  - 3.1.2 Analisis *Microsoft Graph API* . . . 34
  - 3.1.3 Analisis *Slack* . . . 44
  - 3.1.4 Analisis *Heroku* . . . 45
  - 3.1.5 Analisis *Data Flow Diagram* . . . 47

3.2 Analisis Kasus 48

4 Perancangan **49**
4.1 Perancangan Routing Handler 49
4.1.1 *Route get* / 49
4.1.2 *Route get* /authorize 50
4.1.3 *Route get* /slackAuthorize 51
4.1.4 *Route get* /statusChanger 51
4.2 Perancangan *Helper* 52
4.3 Perancangan Basis Data 54

5 Implementasi dan Pengujian **57**
5.1 Implementasi 57
5.1.1 Lingkungan Pengembangan 57
5.1.2 Implementasi Basis Data 58
5.1.3 Implementasi Antarmuka 58
5.2 Pengujian 61
5.2.1 Pengujian Fungsional 61
5.2.2 Pengujian Eksperimental 63

6 Kesimpulan dan Saran **73**
6.1 Kesimpulan 73
6.2 Saran 73

Daftar Referensi **75**

A Kode Program **77**

DAFTAR GAMBAR

1.1 *List* status yang disediakan sebagai pilihan di *Slack*. 2

2.1 Tampilan direktori yang dibuat jika menjalankan express generator. 24

3.1 Diagram alir sistem pada bagian perangkat lunak yang berinteraksi dengan *user*. . 34
3.2 Diagram alir sistem pada bagian perangkat lunak yang dijalankan secara berkala. . 35
3.3 Data kredensial yang diberikan oleh *heroku postgres*. 46
3.4 *Data Flow Diagram level* 0. 47
3.5 *Data Flow Diagram*. 47

4.1 Alur antar *route*. 49
4.2 Rancangan antarmuka halaman awal. 50
4.3 Rancangan antarmuka halaman setelah *login Windows Live*. 50
4.4 Rancangan antarmuka halaman setelah *login Slack*. 51
4.5 ER Diagram untuk tabel Credentials. 54

5.1 Antarmuka halaman awal. 58
5.2 Antarmuka untuk login Windows Live. 58
5.3 Antarmuka untuk memberikan Windows Live izin ke perangkat lunak. 59
5.4 Antarmuka petunjuk login ke Slack. 59
5.5 Antarmuka untuk login Slack. 59
5.6 Antarmuka untuk memberikan Slack izin ke perangkat lunak. 60
5.7 Antarmuka saat selesai melakukan login dan pemberian izin. 60
5.8 62
5.9 Tampilan *Slack* setelah *event* dimulai (Raymond). 62
5.10 63
5.11 63
5.12 Tampilan *Slack* setelah *event* dimulai (Sandy). 64
5.13 64
5.14 Tampilan *Slack* setelah *event* dimulai (Vincent). 65
5.15 65
5.16 Tampilan *Slack* setelah *event* dimulai (Yonathan). 66
5.17 66
5.18 Tampilan *Slack* setelah *event* dimulai (Chris). 66
5.19 67
5.20 Tampilan *Slack* setelah *event* dimulai (Ferdian). 67
5.21 67
5.22 Tampilan *Slack* setelah *event* dimulai (Yehezkiel). 68
5.23 68
5.24 68
5.25 69
5.26 Tampilan *Slack* setelah *event* dimulai (Tedi). 69
5.27 69

5.28 Tampilan *Slack* setelah *event* dimulai (Yosua). 70

DAFTAR TABEL

2.1 Tabel contoh *header* pesan *HTTP* 23

3.1 Tabel *parameter Authorization Code* 36
3.2 Tabel contoh *request Authorization Code* 36
3.3 Tabel contoh *response Authorization Code* 36
3.4 Tabel *parameter response Authorization Code* 37
3.5 Tabel contoh *request Access Token* 37
3.6 Tabel *parameter request Access Token* 38
3.7 Tabel contoh *response Access Token* 38
3.8 Tabel *parameter response Access Token* 39
3.9 Tabel contoh *request call Microsoft Graph* 39
3.10 Tabel contoh *response call Microsoft Graph* 40
3.11 Tabel contoh *request* menggunakan *Refresh Token* 41
3.12 Tabel *parameter request Refresh Token* 41
3.13 Tabel contoh *response* menggunakan *Refresh Token* 41
3.14 Tabel *parameter response Refresh Token* 42
3.15 Tabel contoh *header time zone* 42
3.16 Tabel *parameter header time zone* 42
3.17 Tabel contoh *request event* 43
3.18 Tabel contoh *response event* 43

BAB 1

PENDAHULUAN

1.1 Latar Belakang

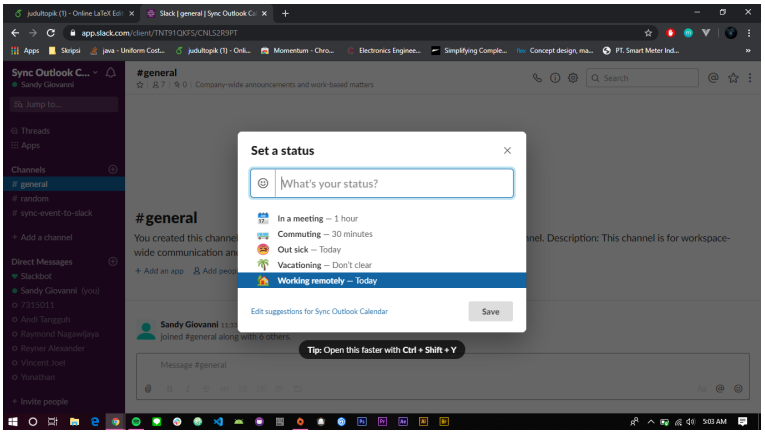
Outlook.com[1] adalah sebuah kumpulan aplikasi berbasis *web* seperti *webmail*, *contacts*, *tasks*, dan *calendar* dari *Microsoft*. Fitur *calendar* sendiri pertama dirilis pada 14 Januari 2008 dengan nama *Windows Live Calendar*. Fitur *calendar* yang dimiliki oleh *Outlook.com Calendar* sendiri memiliki tampilan yang mirip dengan aplikasi kalender *desktop* pada umumnya. Seperti layaknya kalender digital pada umumnya, aplikasi *Outlook.com Calendar* juga bisa menambahkan, menyimpan, dan memodifikasi *event-event* yang dimasukkan oleh pengguna dan bisa dibuka dimana saja karena bersifat *online*.

Slack[2] adalah alat dan layanan kolaborasi tim berbasis *cloud*. *Slack* merupakan singkatan dari "*Searchable Log of All Conversation and Knowledge*". Cara melakukan kolaborasi di aplikasi *Slack* sendiri adalah dengan cara komunitas, grup, atau tim bergabung ke dalam URL yang spesifik yang disebut dengan *workspace*. *Room chat* yang terdapat di dalam *workspace* biasa disebut dengan *Channel*. Ada 2 jenis *channel* di dalam *workspace* yang ada di aplikasi *Slack* yaitu *Public Channel* dan *Private Channel*. Pada *Public Channel*, seluruh anggota dari tim atau komunitas bisa masuk dan bergabung untuk berkomunikasi di *channel* tersebut. Tetapi pada *Private Channel*, hanya anggota yang diizinkan, ditambahkan, dan diundang oleh admin atau pembuat *channel* sajalah yang bisa ikut berkomunikasi di dalam *channel* tersebut. Pada setiap *workspace* dapat ditambahkan dan dipasangkan aplikasi penyokong lainnya seperti contohnya adalah *Trello*, *Github*, *Google Drive*, dan banyak aplikasi lainnya yang bisa digunakan untuk membantu para partisipan di dalam *workspace* tersebut.

Pada *Slack* terdapat status pengguna yang bisa diganti oleh pengguna tersebut untuk menggambarkan keadaan pengguna saat ini. Nilai *default* dari status ini adalah tersedia, tetapi *Slack* sudah menyediakan beberapa pilihan status seperti "*In a meeting*", "*Commuting*", "*Out sick*", "*Vacationing*", dan "*Working remotely*" seperti yang terdapat pada contoh gambar 1.1. Selain itu, *Slack* juga menyediakan pilihan untuk pengguna yang mengisi statusnya sendiri. Dengan adanya fitur status di dalam aplikasi *Slack*, maka hal itu membantu mendeskripsikan keadaan yang sedang dialami oleh pengguna saat itu. Status hanya berupa keterangan di akun pengguna dan tidak berdampak kepada fitur lainnya seperti contohnya fitur obrolan. Yang menjadi latar belakang dirancangnya perangkat lunak ini yaitu terkadang pengguna lupa untuk mengganti status menjadi "*In a meeting*" saat pengguna memiliki jadwal untuk melakukan *meeting*, sehingga status di pengguna masih terlihat tersedia oleh *user* lain yang membuat tidak mengetahui pengguna sedang dalam keadaan *meeting*. Di saat seperti ini, kemungkinan untuk *meeting* terganggu oleh adanya *chat* yang masuk lewat *Slack* pun cukup tinggi. Tetapi status hanya membantu mendeskripsikan keadaan yang sedang dialami oleh pengguna saat itu dan tidak memblokir chat masuk kepada pengguna.

Pada skripsi ini dibuat perangkat lunak yang membaca jadwal dari pengguna yang dicantumkan di aplikasi *Outlook.com Calendar*, lalu akan di integrasikan ke aplikasi *Slack* dengan mengubah status di dalam aplikasi *Slack* sesuai dengan jadwal yang telah didapatkan dari data di *Outlook.com*

[1] https://outlook.live.com/
[2] https://slack.com/



Gambar 1.1: *List* status yang disediakan sebagai pilihan di *Slack*.

Calendar dari pengguna.

Perangkat lunak ini dibuat menggunakan *Node.js*[3] dan memiliki 2 fungsi utama yaitu yang pertama adalah membaca dan mencatat jadwal dari *Outlook.com Calendar* yang membutuhkan adanya *Outlook.com Calendar API*. Lalu fungsi kedua yaitu mengubah status ke aplikasi *Slack* dengan menggunakan *Slack API*. Kedua fungsi dari perangkat lunak ini dijalankan secara berkala sehingga perubahan status yang terjadi pada aplikasi *Slack* tidak terjadi secara *real-time* tetapi berkala setiap 10 menit sekali dilakukan penjalanan dari fungsi perangkat lunak ini.

API atau *Application Programming Interface* adalah sebuah *interface* yang memungkinkan antara 2 aplikasi atau lebih saling berinteraksi. *Outlook Calendar* sendiri memiliki *API* yang berpusat pada *Microsoft Graph API*, lalu pada aplikasi *Slack* juga sudah disediakan *API* sehingga untuk mengkomunikasikan antara kedua aplikasi ini bisa menggunakan *API* dari masing-masing aplikasi yang sudah disediakan.

1.2 Rumusan Masalah

Pada perangkat lunak ini, terdapat rumusan masalah sebagai berikut:

1. Bagaimana cara mendapatkan data *event* dari *Outlook.com Calendar*?
2. Bagaimana mengubah status pada aplikasi *Slack* menggunakan *Slack* API?
3. Bagaimana cara membuat program agar dapat mengubah status pada aplikasi *Slack* di jadwal yang telah didapat dari aplikasi *Outlook.com Calendar*?

1.3 Tujuan

Adapun pada perangkat lunak ini memiliki tujuan sebagai berikut:

1. Mengetahui cara mendapatkan data *event* dari *Outlook.com Calendar*.
2. Mengetahui cara mengubah status pada aplikasi *Slack* menggunakan *Slack* API.
3. Membuat program agar dapat mengubah status pada aplikasi *Slack* di jadwal yang telah didapat dari aplikasi *Outlook.com Calendar*.

[3]https://nodejs.org

1.4 Batasan Masalah

Perancangan perangkat lunak ini dibuat berdasarkan batasan-batasan sebagai berikut:

1. Perangkat lunak ini dijalankan secara berkala sehingga tidak dapat mengubah status secara *real-time*.
2. Hanya berlaku untuk setiap satu pengguna terhubung dengan satu *workspace* saja di dalam *Slack*(Tidak berlaku satu pengguna ke banyak *workspace*).

1.5 Metodologi

Berikut adalah metodologi yang akan digunakan dalam penelitian ini:

1. Melakukan studi literatur tentang *Outlook.com Calendar*, *Slack*, dan juga *Node.js*.
2. Menggunakan aplikasi *Slack* di lingkungan tempat penulis melakukan magang.
3. Melakukan analisis cara melakukan *synchronize* dengan aplikasi *Outlook.com Calendar* secara berkala.
4. Merancang bagian dari perangkat lunak yang akan mengambil data-data *event* dari *Outlook.com Calendar* dan yang bertugas untuk mengubah status pada *Slack* saat waktu sesuai dengan jadwal yang sudah tercatat dari *Outlook.com Calendar*.
5. Mengimplementasi bagian pengambilan data dari *Outlook.com Calendar* dan juga bagian mengatur status pada *Slack* sesuai jadwal yang telah diambil kepada perangkat lunak Integrasi *Outlook.com Calendar* dengan *Slack* serta melakukan pengujian terhadap fitur yang telah diimplementasikan.
6. Penulisan dokumen.

1.6 Sistematika Pembahasan

Setiap bab dalam penelitian ini akan memiliki sistematika pembahasan yang dijelaskan ke dalam poin-poin sebagai berikut:

1. Bab 1: Pendahuluan, yaitu menjelaskan gambaran umum dari penelitian ini yang berisi tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, dan sistematika pembahasan.
2. Bab 2: Dasar Teori, yaitu menjelaskan dan membahas teori yang dibutuhkan untuk melakukan penelitian ini. Meliputi tentang *Outlook.com Calendar API*, *Slack API*, dan *Node.js*.
3. Bab 3: Analisis, yaitu membahas mengenai analisis masalah. Berisi tentang analisis cara pengambilan data dari *Outlook.com Calendar*, dan proses pengubahan status menggunakan program pada *Slack*.
4. Bab 4: Perancangan, yaitu membahas mengenai perancangan aplikasi untuk melakukan sinkronisasi antara *Outlook.com Calendar* dengan *Slack*.
5. Bab 5: Implementasi dan Pengujian, yaitu membahas mengenai implementasi dari perangkat lunak yang telah dirancang dan juga pengujian perangkat lunak tersebut.
6. Bab 6: Kesimpulan dan Saran, yaitu berisi tentang kesimpulan dari penelitian ini dan juga saran yang dapat diberikan untuk penelitian selanjutnya.