Integrasi *Outlook Calender* dan *Slack*

Sandy Giovanni S.–2015730041

1 Data Skripsi

Pembimbing utama/tunggal: **Pascal Alfadian**
Pembimbing pendamping: **-**
Kode Topik : **PAN4505**
Topik ini sudah dikerjakan selama : **1** semester
Pengambilan pertama kali topik ini pada : Semester **46 - Genap 18/19**
Pengambilan pertama kali topik ini di kuliah : **Skripsi 1**
Tipe Laporan : **B** - Dokumen untuk reviewer pada presentasi dan **review Skripsi 1**

2 Latar Belakang

Outlook.com 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 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 komunitas, grup, atau tim bergabunga ke dalam URL yang spesifik. *Room chat* yang terdapat di dalam aplikasi *Slack* biasa disebut dengan *Channel*. Ada 2 jenis *channel* di dalam 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 serta dalam berkomunikasi di dalam *channel* tersebut. *Slack* juga terintegrasi dengan banyak layanan pihak ketiga seperti contohnya adalah *Google Drive*, *Github*, *Trello*, *Dropbox*, dan masih banyak lagi layanan pihak ketiga yang bisa diintegrasikan dengan *Slack* itu sendiri.

Pada *Slack* terdapat status pengguna yang bisa diganti oleh pengguna tersebut untuk menggambarkan keadaan pengguna saat ini. Sebagai *default*, status bisa menggambarkan jika pengguna sedang "*In a meeting*" atau sedang "*Out Sick*", dan banyak status default yang disediakan oleh *Slack*. Serta status pun bisa diisi oleh pengguna secara sendiri sesuai dengan apa yang ingin dituliskan oleh penggunanya. Disinilah 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 sang pengguna sedang dalam keadaan *meeting* yang tidak dapat diganggu. Di saat seperti ini, kemungkinan untuk *meeting* terganggu oleh adanya *chat* yang masuk lewat *Slack* pun cukup tinggi.

Perangkat lunak ini akan dibuat dengan bantuan dari masing-masing API(*Application Programming Interface*). *Outlook Calendar* dan juga *Slack* memiliki API masing- masing yang cara penggunaannya terdapat dalam dokumentasi dari aplikasi tersebut yang bisa ditemui di dalam laman *website* dari masing- masing aplikasi tersebut. Perangkat ini juga akan dibangun menggunakan *Node.js* yang bisa dipelajari melalui laman *website* dokumentasi *Node.js* itu sendiri.

3 Tujuan

Tujuan dari penyusunan skripsi ini antara lain:

- Mengetahui cara menggunakan *Node.js*.
- Mengetahui cara mendapatkan data *event* dari *Outlook Calendar*.
- Mengetahui cara mengubah status pada aplikasi *Slack* menggunakan *Slack* API.
- Mengetahui cara membuat program agar dapat mengubah status pada aplikasi *Slack* di jadwal yang telah didapat dari aplikasi *Outlook Calendar*.

4 Rumusan Masalah

Rumusan masalah pada topik ini adalah:

- Bagaimana cara menggunakan *Node.js*?
- Bagaimana cara mendapatkan data *event* dari *Outlook Calendar*?
- Bagaimana mengubah status pada aplikasi *Slack* menggunakan *Slack* API?
- Bagaimana cara membuat program agar dapat mengubah status pada aplikasi *Slack* di jadwal yang telah didapat dari aplikasi *Outlook Calendar*?

5 Detail Perkembangan Pengerjaan Skripsi

Detail bagian pekerjaan skripsi sesuai dengan rencan kerja/laporan perkembangan terkahir :

1. **Melakukan studi literatur melalui dokumentasi *online* mengenai *Node.js*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Karena banyaknya library yang disediakan oleh Node.js, maka akan dilakukan studi literatur secara terus menerus seiring dengan membuat kode program untuk skripsi ini.

2. **Melakukan studi literatur melalui dokumentasi *online* mengenai aplikasi *Outlook Calendar*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Terdapat perubahan di poin ini dikarenakan adanya aplikasi yang berbeda yaitu aplikasi Outlook Calendar dan Outlook.com Calendar. Aplikasi yang akan digunakan di skripsi ini adalah Outlook.com Calendar yang memiliki API yang berpusat pada Microsoft Graph API. Microsoft Graph API adalah webservice yang berguna untuk mendapatkan data-data yang terdapat di dalam layanan Microsoft 365 yaitu seperti *Azure Active Directory*, layanan *Office* 365 (*SharePoint, OneDrive, Outlook/Exchange, Microsoft Teams, OneNote, Planner,* dan *Excel*), layanan *Enterprise Mobility and Security* (*Identity Manager, Intune, Advanced Threat Analytics,* dan *Advanced Threat Protection*), layanan *Windows* 10 (*activities* dan *devices*), dan *Education*. Terdapat 2 versi referensi untuk *Microsoft Graph API* yaitu versi 1.0 dan juga versi beta, tetapi yang dituliskan pada subbab ini mengacu kepada versi 1.0. Pada versi 1.0, *endpoint* utama yang dipakai adalah mengacu kepada *endpoint https://graph.microsoft.com/v1.0*.

   Untuk menggunakan fungsi dari *Microsoft Graph API*, dibutuhkan untuk mendaftarkan terlebih dahulu aplikasi yang akan dirancang dan memakai fungsi dari *webservice* dari *Microsoft Graph API* ke *Microsoft App Registration Portal*[1]. Pada saat mendaftarkan aplikasinya, pastikan untuk menyalin

[1] https://apps.dev.microsoft.com/

dan menyimpan *application ID* yang adalah pengenal unik untuk aplikasi yang didaftarkan, dan juga menyalin *Redirect URL* yang didaftarkan sebagai *URL* yang akan menerima balikan *authentication* dan juga *token* yang akan dikirim oleh *endpoint Azure AD v2.0*, serta menyalin *application secret* yang didapat saat mengklik "*Generate New Password*" saat mendaftarkan aplikasi (berlaku jika mendaftarkan aplikasi berjenis *web apps*). *Application ID* yang didapat saat mendaftar aplikasi akan dipakai untuk mengisi nilai dari parameter *client_ id* yang akan diisi saat akan melakukan *request* untuk mendapatkan *authorization_ code*. Setelah mendapatkan *authorization_ code*, maka langkah selanjutnya adalah meminta *access_ token* yang membutuhkan *parameter authorization_ code* kepada *field code*, dan juga *application_ secret* yang didapat dari pendaftaran aplikasi sebelumnya yang akan mengisi *field client_ secret*. Response dari request token akan mengembalikkan jangka waktu aktif dari token tersebut dan juga refresh_token yang akan berguna untuk meminta refresh_token saat token sudah *expired*.

Setelah mendapatkan *access token*, barulah layanan untuk mendapatkan data yang tersimpan di *Microsoft* baru bisa diakses dan didapatkan. Ada banyak layanan yang disediakan dari *Microsoft Graph API* yang dikelompokkan menjadi kelas-kelas yang masing-masing memiliki properti dan juga method-method yang cara mengaksesnya memiliki endpoint masing-masing.

3. **Melakukan studi literatur melalui dokumentasi *online* mengenai aplikasi *Slack*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Pada poin ini lebih diutamakan mempelajari aplikasi terutama pada API yang telah disediakan oleh aplikasi Slack. *Slack API* adalah *webservice* yang akan digunakan untuk menghubungkan data yang sudah di dapat dari *Outlook.com Calendar* ke aplikasi *Slack*. Untuk mengakses *Slack API*, kita diharuskan untuk mendaftarkan aplikasi yang akan dibuat dan juga mendaftarkannya ke *workspace* yang akan dipakai untuk menjalankan aplikasi yang akan dibuat. Untuk mendaftarkan aplikasi yang akan dibuat bisa menuju ke laman *https://api.slack.com/apps*. Aplikasi yang sudah terdaftar akan diberikan *Client ID* yang unik dan juga *Client Secret* yang akan digunakan pada proses *OAuth*.

   Proses pertama yang akan dijalani dalam rangkaian proses *OAuth* adalah dengan meminta *authorization code* yang akan berjalan jika aplikasi yang akan dibuat untuk mengarahkan pengguna dan mengirimkan *get request* ke *URL https://slack.com/oauth/authorize* dengan *get parameter* yang wajib yaitu *client_ id* yang diberikan pada saat mendaftarkan aplikasi yang akan dibuat dan juga *get parameter scope*, serta memiliki parameter yang bersifat opsional yaitu *redirect_ uri* yang berfungsi untuk alamat tujuan dari kembalian yang dikirim oleh API, *state* yaitu yang berupa *String* unik untuk diteruskan kembali setelah selesai, dan juga *team* berupa *Slack team ID* dari *workspace* yang berfungsi untuk membatasi. Parameter *scope* disini berguna untuk menentukan dengan tepat bagaimana aplikasi perlu mengakses akun pengguna *Slack*. Penulisan format *parameter scope* yaitu merujuk ke objek yang akan diberi akses, dan dilanjutkan dengan kelas tindakan pada objek yang diberikan izin, contohnya *file:read*. Selain itu ada juga perspektif opsional yang berisi *user*, *bot*, dan *admin* yang akan memengaruhi tindakan yang muncul nantinya di dalam aplikasi *Slack*, contohnya *chat:write:user* yang berarti akan mengirimkan pesan dari pengguna yang memiliki wewenang. Kelas tindakan yang ada disini ada 3 yaitu:

   - **read:** Membaca informasi lengkap dari satu sumber.
   - **write:** Memodifikasi sumber. Bisa melakukan *create*, *edit*, dan *delete* dengan kelas tindakan ini.
   - **history:** Mengakses arsip pesan.

   Authorization code yang telah didapat memiliki waktu kadaluarsa selama 10 menit. Setelah mendapatkan *authorization code*, langkah selanjutnya yang harus dilakukan adalah menukarkan *authorization code* dengan *access token* dengan cara mengirimkan *post request* kepada *endpoint https://slack.com/api/oauth.access*

yang memiliki *request body* yang wajib yaitu *client_ id*, *client_ secret*, dan *code*. *Client_ id* dan juga *client_ secret* didapat dari awal mendaftarkan aplikasi. Parameter *code* didapat dari *authorization code* yang didapatkan dari langkah sebelumnya.

Setelah mendapatkan *access token*, barulah method API yang disediakan bisa dijalankan. Objek yang bisa diakses dan daftar dari method-method API yang disediakan *Slack*.

4. **Melakukan studi literatur melalui dokumentasi *online* mengenai cron dan crontab.**
   Status : Baru ditambahkan pada semester ini.
   Hasil : *Cron* adalah sebuah *daemon* untuk mengeksekusi perintah-perintah yang sudah terjadwalkan. *Daemon* sendiri adalah proses layanan yang berjalan secara *background* dan mengawasi sistem atau menyediakan fungsionalitas untuk proses lainnya. *Cron* dimulai dari */etc/rc.d/init.d* atau */etc/init.d* ketika *sysvinit* digunakan. *File* unit disimpan dan diinstal ke */lib/systemd/system/crond.service* dan *daemon* dimulai dengan cara menjalankan perintah *systemctl start crond.service*. *Cron* mencari ke direktori */var/spool/cron* untuk *file-file crontab*.

   Crontab adalah *file* yang digunakan untuk menjadwalkan eksekusi dari sebuah program. *Crontab* bisa diakses oleh pengguna dengan menjalankan perintah "*crontab*" di *terminal* dilanjutkan dengan perintah yang akan diberikan. Daftar perintah yang bisa dijalankan oleh *crontab* adalah:

   - *crontab -e*
     Command ini digunakan untuk membuat atau mengubah file crontab jika sudah ada.
   - *crontab -l*
     Command ini digunakan untuk menampilkan isi file crontab.
   - *crontab -r*
     Command ini digunakan untuk menghapus file crontab.

   Crontab memiliki baris kode yang berformat "*m h dom mon dow command*" yang memiliki arti:

   - m
     m sebagai menit yang bisa diisi dengan nilai dari 0-59.
   - h
     h sebagai jam yang bisa diisi dengan nilai dari 0-23.
   - dom
     dom sebagai *Day Of Month* (tanggal) yang bisa diisi dengan nilai dari 0-31.
   - mon
     mon sebagai bulan yang bisa diisi dengan nilai dari 0-12.
   - dow
     dow sebagai *Day Of Week* yang bisa diisi dengan nilai dari 0-7 yang menggambarkan hari dalam angka. Nilai 0 dan nilai 7 adalah hari Minggu.
   - command
     command disini berisi program yang akan dijalankan dengan jadwal yang sudah diatur dengan parameter sebelumnya.

   Semua nilai dari *parameter* m, h, dom, mon, dan dow bisa diisi dengan simbol bintang (*) yang memiliki arti dijalankan setiap menit, setiap jam, setiap hari, setiap bulan tergantung dari posisi dimana simbol itu ditempatkan.

5. **Melakukan analisis cara melakukan *synchronize* dengan aplikasi *Outlook Calendar* secara berkala.**

Status : Ada sejak rencana kerja skripsi.
Hasil : Analisis ini dilakukan harus dengan cara membaginya menjadi analisis atas pemakaian Microsoft Graph API, dan juga analisis mengenai Cron dan Crontab.

6. **Melakukan analisis cara aplikasi merespons ketika menjadwalkan perubahan status di aplikasi *Slack*, dengan kemungkinan masalah seperti : ada *event* yang baru ditambahkan setelah program melakukan *synchronize* secara berkala atau ada *event* yang beririsan dengan *event* lainnya sehingga terdapat masalah menentukan kapan status dibuang/ dikembalikan lagi ke status semula.**
   Status : belum dikerjakan.
   Hasil :

7. **Merancang bagian dari perangkat lunak yang akan mengambil data-data *event* dari *Outlook Calendar*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : berdasarkan analisis singkat, tidak dilakukan analisis lebih jauh karena tidak diperlukan struktur data baru, karena sudah disediakan oleh OpenSteer versi terbaru

8. **Merancang bagian dari perangkat lunak yang bertugas untuk mengubah status pada *Slack* saat waktu sesuai dengan jadwal yang sudah tercatat dari *Outlook Calendar*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

9. **Mengimplementasi bagian pengambilan data dari *Outlook Calendar* dan juga bagian mengatur status pada *Slack* sesuai jadwal yang telah diambil kepada perangkat lunak Integrasi *Outlook Calendar* dengan *Slack*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

10. **Melakukan pengujian terhadap fitur yang telah diimplementasi.**
    Status : Ada sejak rencana kerja skripsi.
    Hasil :

11. **Menulis dokumen skripsi**
    Status : Ada sejak rencana kerja skripsi.
    Hasil : Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

6 Pencapaian Rencana Kerja

Langkah-langkah kerja yang berhasil diselesaikan dalam Skripsi 1 ini adalah sebagai berikut:

1. Melakukan studi literatur melalui dokumentasi *online* mengenai aplikasi *Outlook Calendar*.
2. Melakukan studi literatur melalui dokumentasi *online* mengenai aplikasi *Slack*.

3. Melakukan studi literatur melalui dokumentasi *online* mengenai cron dan crontab.

7 Kendala yang Dihadapi

Kendala - kendala yang dihadapi selama mengerjakan skripsi :

- Sulitnya fokus untuk mengerjakan.
- Terlalu banyak godaan berupa hiburan (game, film, dll).
- Dokumentasi dari Outlook Calendar kurang jelas dan ada beberapa versi dokumentasi mengacu kepada versi API yang disediakan.

Bandung, 01/05/2019

Sandy Giovanni S.

Menyetujui,

Nama: Pascal Alfadian
Pembimbing Tunggal