BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Organisasi

Kerja magang dilaksanakan pada perusahaan PT. SMART Tbk. ditempatkan dalam divisi IT pada tim *digital innovation*. Tim ini, mempunyai tugas untuk membuat rpa dan aplikasi web yang dibutuhkan oleh tim lain. Tim ini juga mempunyai 4 bidang pemisah, yaitu *web application developer*, *support*, *RPA development* dan *product analyst*. Selama kerja magang dilaksanakan, bimbingan akan dilakukan oleh Ibu Melisa dan Bapak Faris Salman Hakim, selaku *web application developer* pada tim *digital innovation*. Selain itu, projek akan ditugaskan kepada tim *digital innovation* dengan melalui ketua dari tim *digital innovation*, yaitu bapak Arif Khairiansyah.

3.2 Tugas yang Dilakukan

Tugas yang dilakukan pengembang atau *developer* adalah mengembangkan situs web sesuai dengan persyaratan yang dikumpulkan oleh analis produk atau *product analyst*. Sebelum proses pengembangan dimulai, Product Analyst terlebih dahulu mengumpulkan persyaratan aplikasi yang dibutuhkan oleh pelanggan atau pengguna. Analis produk kemudian membuat prototipe persyaratan , yang menjadi desain situs web. Prototipe wireframe diberikan kepada pengguna. Jika pengguna ingin menambahkan persyaratan apa pun, analis produk akan menambahkan wireframe sesuai dengan kebutuhan pengguna. Setelah wireframe disetujui oleh pengguna, analis produk membuat jadwal proyek. Garis waktu yang dibuat oleh analis produk dikomunikasikan kepada pengguna sehingga mereka tahu kapan harus mulai menggunakan situs web untuk mendukung pekerjaan mereka. Setelah semuanya siap dan pengguna menyetujuinya, pengembangan dimulai dan pengembang dapat mulai bekerja. Pada minggu pertama, pengembang mulai merancang database sesuai dengan persyaratan wireframe yang dibuat oleh analis produk . Setelah desain *database* selesai, pengembangan *frontend* dan juga *backend* akan dimulai. Apabila pengembangan *website* sudah berjalan 50% (berdasarkan dengan timeline yang telah ditentukan) akan diberlakukan *mid review*. *Mid review* dilakukan dengan pengguna agar mereka tetap mendapat informasi tentang kemajuan pengembangan situs web. Setelah pengembangan mencapai 100% (sesuai

jadwal yang ditetapkan), tinjauan akhir dilakukan dengan pengguna menggunakan fungsi yang sama dengan tinjauan jangka menengah, yaitu memberi tahu pengguna tentang kemajuan yang dicapai selama proses pengembangan. Setelah validasi akhir, pengguna menyelesaikan UAT (User Acceptance Test). Menurut UAT, error biasanya ditemukan pada website atau pengguna ingin menambahkan persyaratan (persyaratan baru pada website yang mereka kembangkan). Setelah UAT selesai, pengembang mempunyai waktu untuk memperbaiki bug atau menambahkan fitur berdasarkan persyaratan tambahan dari hasil UAT. Setelah semua perbaikan dan penyempurnaan kesalahan berdasarkan persyaratan tambahan selesai, penerapan dimulai. Penerapan memindahkan situs web dari lingkungan lokal (lingkungan yang digunakan oleh pengembang selama pengembangan) ke online atau langsung. Server dipindahkan agar pengguna dapat mengaksesnya. Bukan hanya pengembang atau pengguna internal.

Tugas yang dilakukan pada proses magang ini adalah melakukan pengembangan program. Program yang dikembangkan adalah program yang akan digunakan oleh aplikasi yang ada pada TV Samsung. Dalam program tersebut, akan ada proses - proses yang akan melakukan screenshot pada *dashboard* - *dashboard* yang akan ditampilkan pada TV. Proses tersebut akan berjalan dengan jadwal yang telah ditentukan. Terdapat 65 total *dashboard* yang harus di-*screenshot* oleh program. Dari program tersebut, terdapat *application programming interface*(API) yang bisa dipanggil oleh aplikasi yang ada pada TV. API tersebut akan mengembalikan gambar - gambar *dashboard* untuk ditampilkan pada TV tersebut. Selain menampilkan *dashboard*, TV juga harus menampilkan video. Video juga ditampilkan sesuai dengan jadwal yang telah ditentukan. Video juga perlu didapatkan melalui API yang telah dibuat. API yang telah dibuat akan mengambil video - video yang telah didapatkan pada *Network Attached Storage*.

3.3 Projek - Projek yang Dikerjakan Selama Pelaksanaan Magang

3.4 Uraian Pelaksanaan Magang

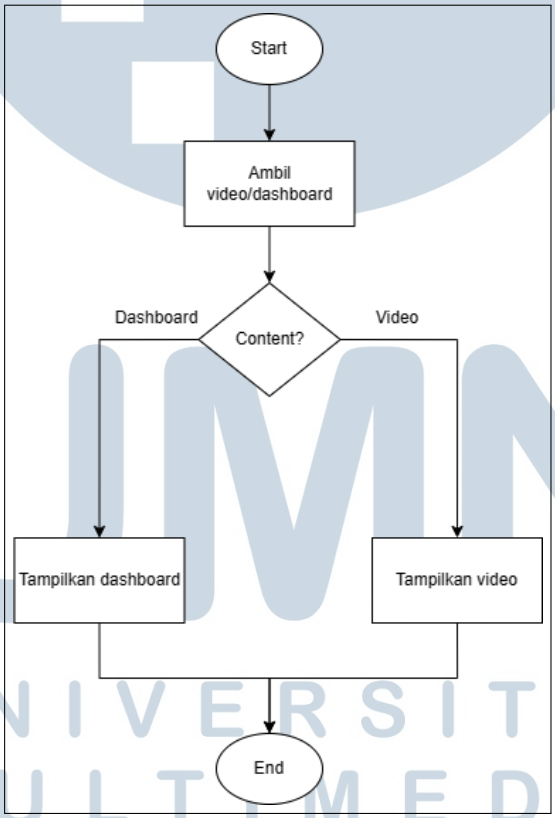
Tugas pengembang untuk melakukan pekerjaan pemeliharaan dan pengembangan situs web. Setiap pengembang akan mengerjakan setidaknya satu proyek dalam satu waktu. Proyek ini akan memiliki waktu 2-6 bulan untuk dilaksanakan. Waktu ini berbeda-beda bergantung pada kompleksitas program yang dikembangkan. Tugas yang dilakukan sebagai bagian dari magang ini adalah

proyek pengembangan API.

Selama pelaksanaan magang pada 6 bulan ini, ada total 4 projek yang dikerjakan. Berikut adalah uraian dari projek:

3.4.1 TizenOS

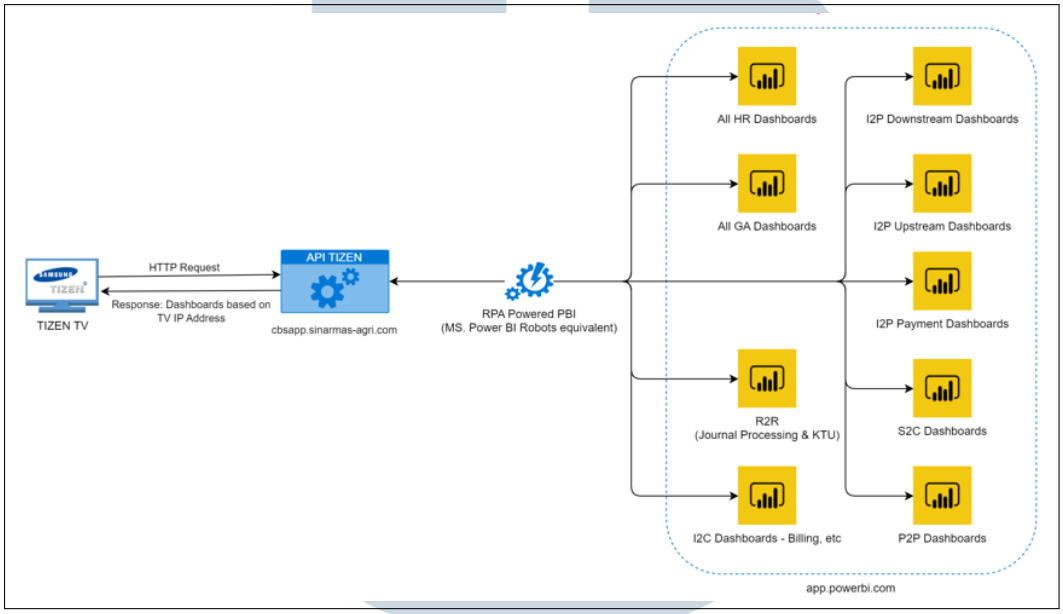
Projek TizenOS adalah aplikasi yang menjalani proses dengan menggunakan NodeJS untuk meng-*capture* dashboard yang ada pada *website* PowerBI untuk keperluan Tizen. Projek Tizen ini, bertujuan untuk menampilkan *dashboard - dashboard* divisi di setiap TV. Proses yang perlu dilakukan adalah program akan melakukan proses secara otomatis menggunakan *scheduler*. Proses tersebut akan melakukan *web crawling* untuk *screeenshot dashboard* yang akan ditampilkan di TV.



Gambar 3.1. Proses diagram aplikasi tizen

Gambar 3.1 merupakan gambaran besar proses yang akan dilakukan oleh program. Pertama, program yang ada pada TV akan memanggil API. API tersebut akan menyediakan gambar - gambar yang dibutuhkan oleh TV tersebut dalam bentuk *file* base64. Selain menampilkan gambar, aplikasi di TV juga dapat

menampilkan video. Video ini, didapatkan melalui *Network Attached Storage* (NAS). Video akan ditampilkan dengan cara menampilkan API, lalu API akan mengembalikan *link* dari video, kemudian program TV akan menampilkan video tersebut.

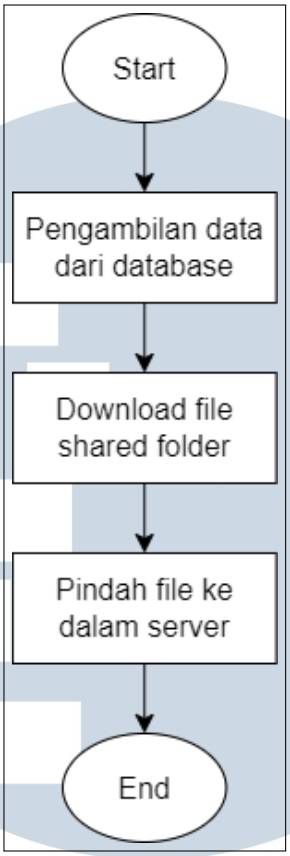


Gambar 3.2. Proses diagram aplikasi tizen

Gambar 3.2, merupakan *flow diagram* dari program yang ada pada API. Pada program ini, akan ada *scheduler* yang berfungsi untuk menjadwalkan pengambilan gambar *dashboard* pada *website* powerbi. Jadwal mengambil *dashboard* adalah setiap hari, dan program ini akan meng-*capture* sebanyak 65 *dashboard* setiap harinya.

3.4.2 NPWP Update

Program yang dibuat dalam projek ini bertujuan untuk melakukan pemindahan *file* NPWP bagi *vendor*, karena adanya pembaruan *file* NPWP dari pemerintah maka NPWP harus di-data kembali. Proses yang harus dilakukan adalah pertama vendor akan melakukan pengisian form terkait NPWP dan meng-*upload file* NPWP. Kemudian, *file* akan dimasukkan ke dalam *shared folder* atau *cloud storage*, dari *shared folder* tersebut, program akan melakukan penarikan *file* - *file* yang telah di-*upload*. *File* kemudian akan dipindahkan ke dalam *server*. Berikut adalah *flow diagram* dari program.



Gambar 3.3. Proses diagram aplikasi NPWP Update

Program akan berjalan dengan jadwal yang ditentukan menggunakan *scheduler*. Pertama, program akan mengambil data dalam *database*. *Database* ini, berisi data - data yang telah diisi oleh vendor dan nama *file* dari *file* yang telah di-*upload* oleh pengisi form. Setelah mengambil data, data tersebut kemudian akan dipakai untuk men-*download* file NPWP. Setelah men-*download file* akan dipindahkan ke dalam server sehingga dapat diakses oleh pengguna.

3.4.3 Project Documentation

Project documentation atau dokumentasi projek, merupakan projek yang dilaksanakan untuk mencatat semua *API endpoint* yang ada pada projek - projek yang telah berjalan. Fungsi dari dokumentasi projek ini adalah mempermudah *developer* - *developer* baru dalam mengerjakan projek yang telah berjalan atau *existing projects*. Projek ini menggunakan *software* Swagger. Swagger adalah *tools* untuk melakukan dokumentasi API.

3.4.4 Tzu Chi Sinarmas Website Activity Module

Pada projek ini, tzu chi sinarmas ingin melakukan pengembangan pada *website* yang telah ada, yaitu website app.tzuchi.sinarmas.com. Pada modul - modul sebelumnya, website ini berfungsi untuk membantu karyawan tzu chi sinarmas ataupun sinarmas untuk melakukan donasi kepada tzu chi sinarmas. Dan pada projek ini, tzu chi sinarmas ingin menambahkan modul pada *website*-nya. Modul ini, berfungsi untuk membantu tzu chi sinarmas untuk melakukan proses pengajuan kegiatan - kegiatan, seperti proses permintaan tanda tangan, persetujuan dan *monitoring reimbursement*. Peran pengembang dalam projek ini adalah untuk mengembangkan halaman - halaman dan fitur - fitur yang dibutuhkan oleh tzu chi sinarmas dalam *website*-nya.

Berikut merupakan tabel pekerjaan yang dilakukan setiap minggunya selama pelaksanaan kerja magang.

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	*Maintain website* tzu chi sinarmas
2	*Bug fixing website* tzu chi sinarmas
3	Mengerjakan halaman *access control* pada *website* tzu chi sinarmas
4	Mengerjakan halaman *user role* pada *website* tzu chi sinarmas
5	Melakukan dokumentasi API pada projek *budget automation* menggunakan SwaggerJS
6	Mengerjakan tampilan *website* tzu chi Sinarmas halaman *user role*
7	Membuat proses untuk meng-*capture dashboard* - *dashboard* pada *website* PowerBI
8	Membuat API untuk menampilkan *dashboard* sesuai dengan IP address dari TV
9	Membuat API untuk projek *distributor dashboard*
10	Membuat dokumentasi API pada projek *Claim Tax Refund*
11	Melakukan UAT *(User Acceptance Test)* untuk projek *distributor dashboard*
12	Bug fixing API pada project TizenOS
13	Membuat API untuk projek NPWP Update
14	Melakukan perubahan *code* pada projek *distributor dashboard*

3.4.5 Minggu Pertama: *Maintain* website tzu chi Sinarmas

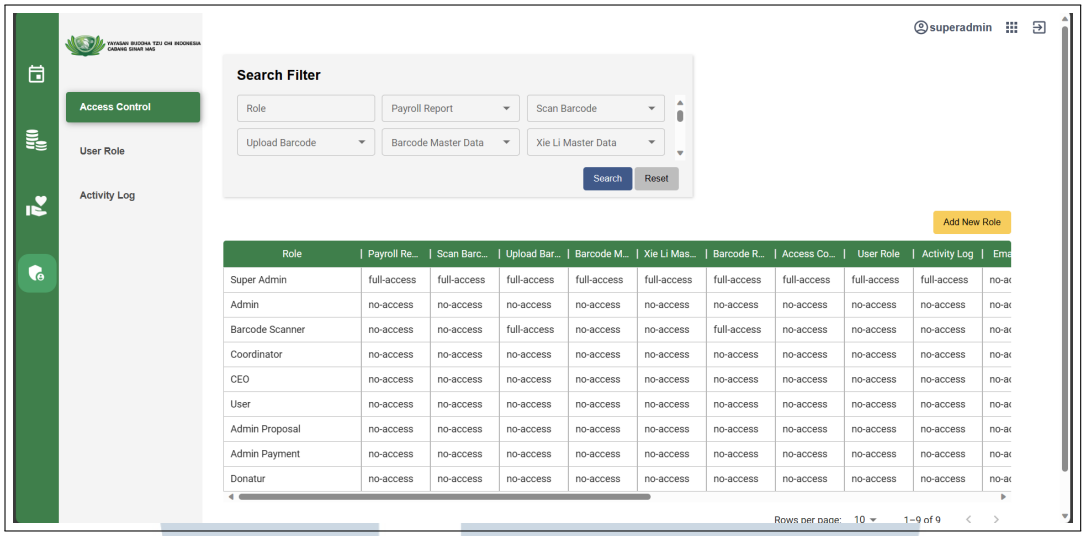
Pada minggu ini, dilakukan *maintaining* pada website tzu chi sinarmas. Website ini di-*maintain* karena baru melakukan *deployment* pada minggu sebelumnya. Tujuan dari *maintaining* ini adalah untuk memastikan bahwa *website* yang telah dilakukan *deployment* tidak memiliki kendala pada *server* ataupun dalam *client side*-nya. Proses *maintaining* yang dilakukan adalah melakukan perbaikan - perbaikan pada database seperti melakukan *cleansing* pada data, sehingga mendapatkan data yang bisa dipakai oleh *website* seperti yang telah didesain.

3.4.6 Minggu Kedua: *Bug fixing website* tzu chi sinarmas

Bug fixing berdasarkan *feedback* yang telah didapatkan pada minggu sebelumnya. Akan dilakukan perbaikan pada *code* berdasarkan *feedback* pengguna terkait *website*. Perbaikan yang dilakukan adalah tampilan ataupun secara data. *Bug fixing* yang dilakukan adalah berkaitan dengan fitur email yang ada pada *website* tzu chi. Kendala yang terjadi adalah email tersebut tidak terkirim kepada pengguna sehingga pengguna tidak bisa melanjutkan prosesnya. Setelah *bug fixing* selesai, *website* akan kemudian dilakukan *deployment* kembali sehingga dapat di-*test* oleh *user*.

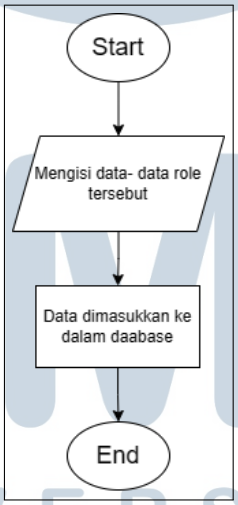
3.4.7 Minggu Ketiga: Mengerjakan halaman *access control* pada *website* tzu chi sinarmas

Pada minggu ketiga, pengerjaan halaman *access control*. Sama seperti halaman - halaman lainnya, halaman *access control*, dikembangkan dengan menggunakan *framework* ReactJS dan NodeJS. Halaman *access control* ini, mempunyai kegunaan untuk menentukan akses - akses dari *role* pada akun yang terdaftar pada *website* tzu chi sinarmas. Fungsi dari *role* adalah untuk mengetahui apakah *role* tersebut mempunyai akses pada halaman yang ada pada *website* tzu chi sinarmas, setiap akun pada website ini mempunyai *role*-nya masing - masing. Berikut adalah tampilan dari halaman *access control*.



Gambar 3.4. Halaman *Access Control Website* Tzu Chi Sinarmas

Terlihat tabel pada gambar 3.4, tabel tersebut berisi nama *role* dan akses pada halaman - halaman yang ada pada website tzu chi sinarmas. Pengguna juga dapat melakukan penambahan pada *role*. Berikut merupakan flow diagram untuk penambahan *role*.

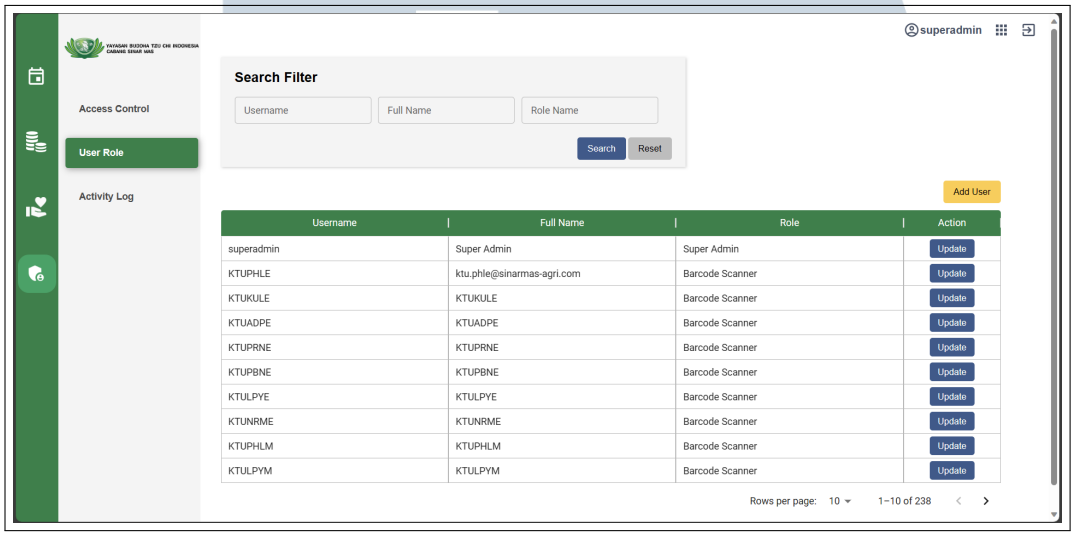


Gambar 3.5. Flow Diagram Add User Role

Sesuai dengan *flow diagram* di atas, pengguna akan melakukan pengisian data - data yang dibutuhkan untuk menambahkan *role*. Data - data yang dibutuhkan untuk penambahan *role* adalah nama *role* dan akses - akses pada halaman yang ada.

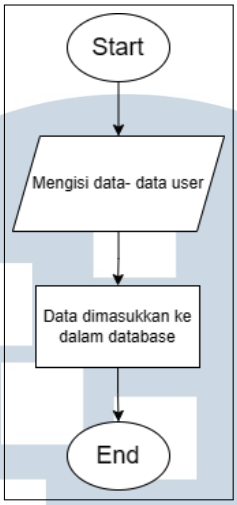
3.4.8 Minggu Keempat: Mengerjakan halaman *user role* pada *website* tzu chi sinarmas

Pada halaman *user role*, pengguna *website* akan bisa menentukan *role* dari setiap akun. Setiap akun mempunyai satu *role*. *Role* tersebut kemudian menentukan halaman apa saja yang dapat diakses oleh akun tersebut. Berikut merupakan tampilan dari halaman *user role*.



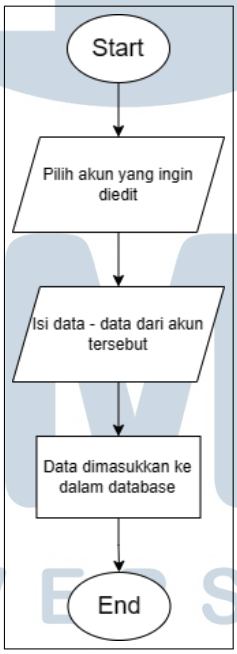
Gambar 3.6. Tampilan Halaman User Role

Tabel yang ada pada halaman tersebut merupakan akun - akun yang terdaftar pada *website* tzu chi sinarmas. Selain itu, terdapat juga *role* dari pengguna tersebut. Pengguna dapat melakukan penambahan akun dan juga penggantian *role* dari akun. Berikut merupakan flow diagram dari penambahan akun dan penggantian *role* dari akun.



Gambar 3.7. Flow Diagram Add Akun

Pada proses ini, pengguna akan diminta untuk mengisi data - data yang dibutuhkan untuk membuat akun, seperti *username*, nama, *password* (akan diganti saat akun pertama kali login), dan *role*.

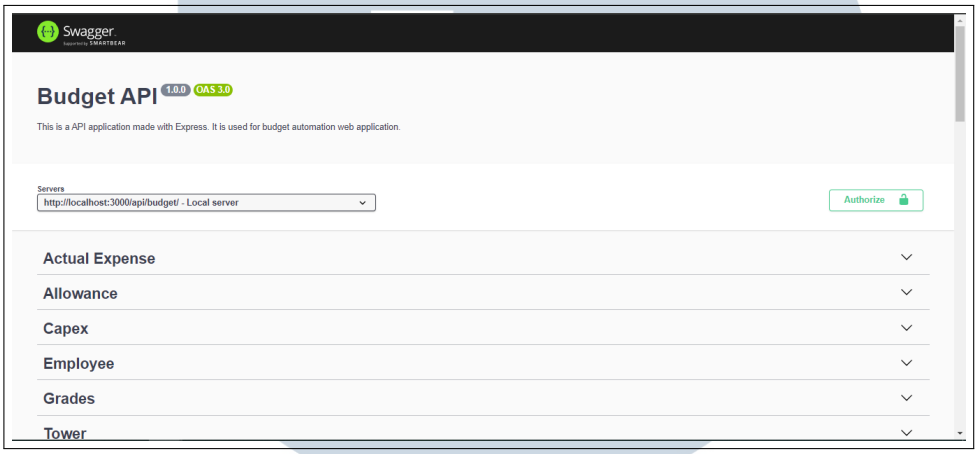


Gambar 3.8. Flow Diagram Peng-editan Akun

Pada saat meng-*edit* data dari akun, pengguna akan memilih akun yang ingin di-*edit* terlebih dahulu, kemudian melakukan pengisian data yaitu *username*, nama, dan *role*.

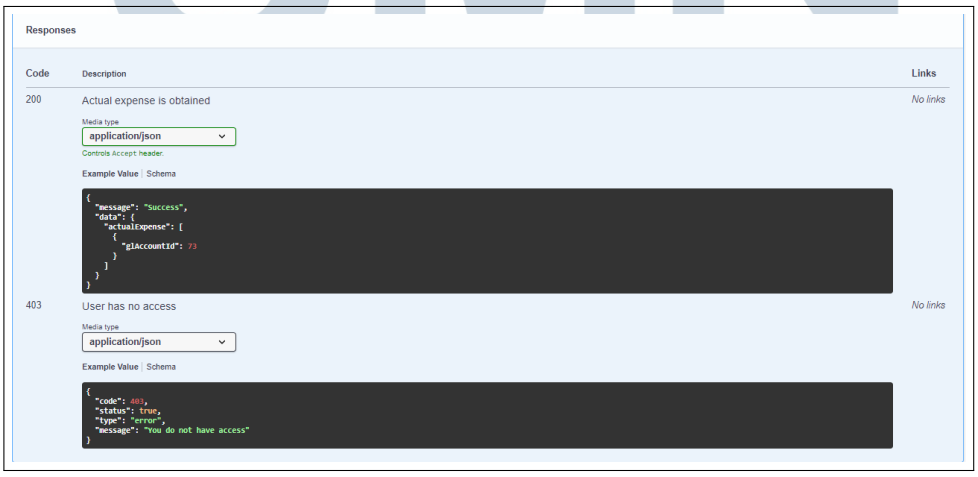
3.4.9 Minggu Kelima: Melakukan dokumentasi API pada projek *budget automation* menggunakan SwaggerJS

Swagger adalah *library* yang bisa digunakan dalam NodeJS untuk melakukan dokumentasi pada API. Swagger sudah menyediakan UI secara otomatis sehingga *developer* tidak perlu membuat *front end* untuk dokumentasi. Berikut merupakan tampilan dari Swagger.



Gambar 3.9. Tampilan halaman swagger

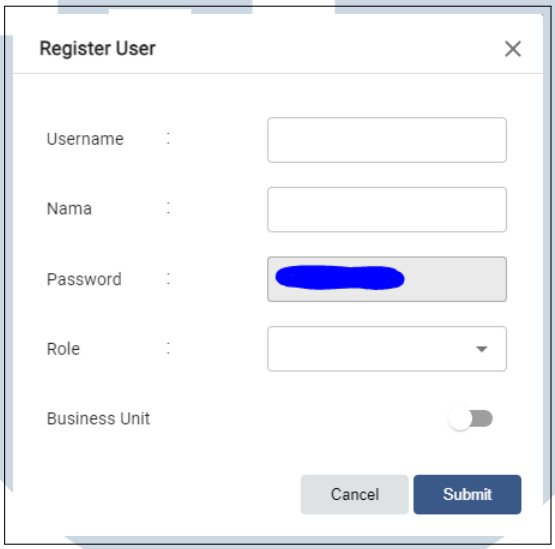
Pada gambar 3.9, terdapat *general info* dan *grouping* dari API - API yang ada pada projek *budget automation*. Pengguna juga dapat melihat *response* dari API yang dipanggil tersebut seperti pada gambar 3.10. Pada API yang ada pada gambar 3.10, ada 2 *response* yang mungkin dikembalikan oleh API, yaitu *response* 200 (sukses beserta data - data yang diminta) dan *response* 403 (pengguna tidak memiliki akses untuk memanggil API tersebut)



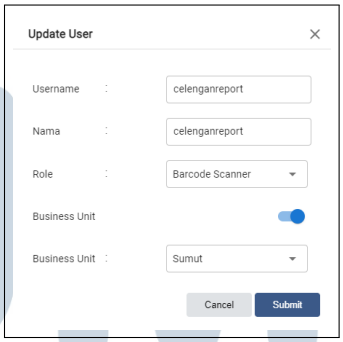
Gambar 3.10. Tampilan halaman swagger

3.4.10 Minggu Keenam: Mengerjakan tampilan *website* tzu chi sinarmas halaman *user role*

Melanjutkan tampilan halaman *user role* pada *website* tzu chi sinarmas. Dengan menambahkan *modal* untuk meng-*update* dan menambahkan akun.



Gambar 3.11. Tampilan penambahan akun



Gambar 3.12. Tampilan *edit* akun

Form yang ada pada penambahan akun dan peng-editan akun memiliki *field* yang sama, kecuali *password*, karena password tidak bisa diubah melalui halaman *user role* tersebut.

3.4.11 Minggu Ketujuh: Membuat proses untuk meng-*capture dashboard* - *dashboard* pada *website* PowerBI

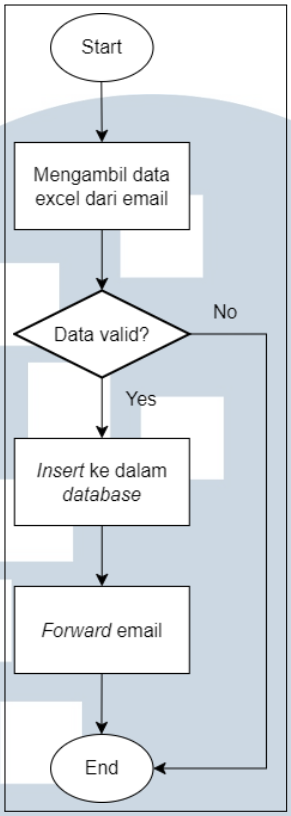
Pada minggu ketujuh, membuat proses - proses dan API yang telah ditentukan, dengan menggunakan NodeJS, RPA, dan aplikasi Tizen.

3.4.12 Minggu Kedelapan: Membuat API untuk menampilkan *dashboard* sesuai dengan IP address dari TV

Pada minggu ini, akan membuat API untuk menampilkan *dashboard* sesuai dengan IP address TV. IP address, akan berguna sebagai *identifier* dari TV karena banyaknya TV yang harus menampilkan *dashboard* dalam waktu yang bersamaan. Sebanyak 2 TV di setiap lantai, yaitu 4 lantai. Di setiap TV tersebut, akan ada *dashboard* - *dashboard* yang berbeda yang harus ditampilkan. *Dashboard* yang ditampilkan merupakan *dashboard* divisi sesuai dengan letak TV.

3.4.13 Minggu Kesembilan: Membuat API untuk projek *distributor dashboard*

Pada projek ini, API dibutuhkan untuk melakukan pembacaan data dari microsoft excel, memasukkan data tersebut ke dalam *database*, dan mengirim email kepada pengguna. Program ini akan berjalan secara otomatis dengan menggunakan *scheduler*. Data excel akan didapatkan melalui email yang akan dikirimkan oleh pengguna setiap harinya. Lalu program akan bertugas untuk mengambil excel yang telah dikirimkan, lalu program akan melakukan proses untuk memasukkan data ke dalam database setelah memvalidasi data tersebut. Setelah dimasukkan ke database, program akan melakukan *forward* email ke pengguna. Berikut merupakan flow diagram dari program.



Gambar 3.13. Proses diagram aplikasi *distributor dashboard*

API pada projek *distributor dashboard* akan dikembangkan menggunakan NodeJS dan python. NodeJS akan berguna untuk melakukan proses - proses dari program yang mempunyai kaitan dengan *database*. Sedangkan python, akan berfungsi untuk membaca data pada excel yang ada pada email. Python akan kemudian mengirim data tersebut dan ke dalam NodeJS sehingga NodeJS dapat memasukkan data yang telah diproses oleh python ke dalam *database*.

3.4.14 Minggu Kesepuluh: Membuat dokumentasi API pada projek *Claim Tax Refund*

Sama dengan minggu sebelumnya, pada minggu ini, akan membuat dokumentasi API pada projek *claim tax refund* dengan menggunakan SwaggerJS. Dokumentasi ini berguna untuk yang ingin melakukan *bug fixing* pada projek ini. Selain itu, dokumentasi ini juga berguna untuk melakukan *debugging* sehingga lebih mudah untuk melihat *response* apa yang diberi oleh API yang ada pada projek ini. Dokumentasi yang dibuat pada minggu ini, mempunyai tampilan yang sama seperti minggu kelima, perbedaannya adalah API - API yang ada pada listnya dan *response* yang dikembalikan oleh API tersebut.

3.4.15 Minggu Kesebelas: Melakukan UAT *(User Acceptance Test)* untuk projek *distributor dashboard*

Pada minggu kesebelas, UAT untuk projek *distributor dashboard* sedang dilakukan. UAT berguna untuk memastikan apakah program telah berjalan sesuai dengan ekspetasi pengguna. UAT melibatkan pengguna dan meminta pengguna program untuk melakukan testing. Apabila terdapat proses yang tidak sesuai dengan ekspetasi pengguna, pengguna dapat memberi tahu pihak *product analyst* untuk mendapatkan *assessment* dari *product analyst*. *Product analyst* kemudian akan melakukan pengecekan apakah permintaan tersebut telah ditentukan pada awal projek atau tidak. Apabila permintaan diterima oleh *product analyst*, maka akan dilakukan perubahan pada program. Setelah program selesai diubah, pengguna akan melakukan *testing* kembali. Ketika sudah sesuai dengan ekspetasi pengguna, pengguna akan melakukan *sign off*, atau tanda tangan bahwa program telah sesuai dengan ekspetasi dan telah berjalan dengan lancar.

3.4.16 Minggu Keduabelas: *Bug fixing* API pada projek TizenOS

Melakukan *bug fixing* terkait projek Tizen, dengan melakukan optimisasi pada program dalam melakukan *screenshot* dashboard sehingga tidak meng-*screenshot* hal yang salah karena adanya *bug* pada program saat melakukan *screenshot*. Kemudian melakukan *fixing* terhadap *dashboard - dashboard* yang ditampilkan agar TV dapat menampilan *dashboard* dengan benar sesuai per-divisi.

3.4.17 Minggu Ketigabelas: Membuat API untuk projek NPWP Update

Sama seperti API - API sebelumnya, API projek ini akan dikembangkan menggunakan NodeJS. NodeJS akan digunakan untuk membaca data dan memproses data tersebut. NodeJS juga akan melakukan *download* dari *shared folder*.

3.4.18 Minggu Keempatbelas: Melakukan perubahan *code* pada projek *distributor dashboard*

Perubahan pada *code distributor dashboard* dilakukan karena adanya perubahan *requirement minor* atau kecil pada program. Maka dari itu, program harus dilakukan perubahan pada *code*. Setelah melakukan perubahan pada program,

akan kembali dilakukan *deployment* sehingga bisa dipakai kembali oleh pengguna. Perubahan *requirement* tersebut melibatkan pemrosesan data yang ada pada email dan proses *forward email*. Karena adanya perubahan tersebut, proses yang ada pada *script* python harus di-*refactor* sehingga memenuhi kebutuhan baru tersebut.

3.5 Kendala yang Ditemukan

3.5.1 Kendala yang ditemukan

- Susah menyesuaikan *styling website* pada desain yang telah ditentukan.
- *Requirement* yang berganti pada saat setelah *development* sehingga menimbulkan *bugs* karna mengejar *timeline* yang ditentukan.

3.5.2 Solusi yang ditemukan

- Membaca dokumentasi yang telah disediakan oleh *tools* tersebut. Pada kasus ini, Material UI telah menyediakan dokumentasi untuk menyesuaikan desain sesuai apa yang diinginkan.
- Memikirkan semua kondisi - kondisi yang mungkin terjadi seperti validasi - validasi yang diperlukan sebelum menjalani proses *development*.