BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Organisasi

Selama pelaksanaan kegiatan magang di PT Kalbe Farma, posisi magang yang ditempati merupakan *Web Developer* pada bagian *Technical Support*. *Web Developer* menyangkut pembuatan *website* pada bagian *frontend* serta *backend*. Posisi ini terdiri dari 3 orang magang yang ketiganya berasal dari program *MSIB*. *Jobdesk* setiap orang berbeda beda, dan pada proyek ini *jobdesk* yang didapat merupakan *back end*.

Proyek JPM ini dikepalai oleh *supervisor* dari *Technical Support* yaitu Bapak Anton Suprayudi yang membantu menerjemahkan kebutuhan pihak produksi menjadi suatu perancangan serta menjadi penengah antara *developer* dengan pihak produksi.

3.2 Tugas yang Dilakukan

Selama pelaksanaan kerja magang sebagai *Web Developer* tugas yang dikerjakan meliputi:

- Merancang desain *website* JPM
- Merancang struktur *database website* JPM
- Membuat *backend website* JPM

3.3 Uraian Pelaksanaan 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	Memahami *website* yang akan digunakan oleh perusahaan serta pendalaman *stack MERN*
2	Meng-*install tools* yang dibutuhkan dan memulai pembuatan prototipe menggunakan *figma* proyek SOP
3	Menyesuaikan *figma* SOP sesuai dengan permintaan serta perubahan yang ada.
4	Menyesuaikan *figma* SOP serta pemberitahuan pergantian proyek menjadi proyek JPM
5	Membuat *figma* JPM sesuai dengan permintaan
6	Menyesuaikan *figma* JPM dengan permintaan serta perubahan yang ada.
7	Memulai pengerjaan *backend website* untuk proyek JPM.
8	Melanjutkan pengerjaan *backend website* untuk proyek JPM.
9	Menyesuaikan *backend website* sesuai dengan permintaan.
10	Melanjutkan dan melakukan revisi terhadap *backend website*
11	Melanjutkan dan melakukan revisi terhadap *backend website*
12	Melanjutkan dan melakukan revisi terhadap *backend website*
13	Melanjutkan dan melakukan revisi terhadap *backend website*
14	Melakukan perpindahan *website* dari *stack MERN* menjadi *Laravel*
15	Melanjutkan dan melakukan revisi terhadap *backend website Laravel*
16	Melanjutkan dan melakukan revisi terhadap *backend website Laravel*
17	Melanjutkan dan melakukan revisi terhadap *backend website Laravel*

Pada minggu pertama, dilakukan pengenalan kantor dan supervisi serta adanya kegiatan *onboarding*. Selain itu, supervisi juga menjelaskan proyek yang akan dikerjakan nantinya yaitu proyek SOP. Supervisi juga menjelaskan mengenai *stack* yang digunakan. *Stack* yang digunakan adalah *stack* MERN yang berupa singkatan dari *MongoDB, Express.js, React.js, Node.js*. Supervisi juga menyarankan untuk mendalami *stack MERN* terlebih dahulu sebelum dimulainya proyek.

Pada minggu ke-dua, supervisi menjelaskan *tools-tools* yang perlu di-*install* dan menjelaskan kegunaan serta alasan digunakannya supaya disamakan versinya dengan yang sudah dipakai sebelumnya di perusahaan. Sembari menunggu peng-*install*-an, prototipe untuk proyek mulai dibuat.

Pada minggu ke-tiga, prototipe yang dibuat masih belum sempurna dan supervisi mendapatkan umpan balik untuk mengubah prototipe tersebut supaya lebih sesuai dengan spesifikasi kantor. Perubahan ini meliputi penambahan fitur dan modifikasi terhadap fitur yang kurang efisien.

Pada minggu ke-empat prototipe mencapai tahap akhir dan hanya perlu dilakukan penyelesaian, tetapi karena adanya sedikit masalah menyangkut aplikasi yang mengharuskan adanya perubahan signifikan terhadap aplikasi serta lingkungan kerja yang ada, akhirnya diputuskan agar prototipe tersebut sementara dihentikan dan supervisi memberikan proyek baru yaitu proyek JPM.

Pada minggu ke-lima, prototipe untuk *website* JPM mulai dibuat sesuai dengan permintaan. Prototipe yang dibuat disesuaikan dengan perubahan dan penambahan fitur yang dilakukan selama satu minggu.

Pada minggu ke-enam, prototipe untuk *website* JPM masih dirubah dan disesuaikan dengan permintaan. Ada pula penambahan fitur yang signifikan pada minggu keenam yang menyebabkan terjadinya perubahan desain pada prototipe.

Pada minggu ke-tujuh, prototipe sudah selesai sepenuhnya dan mendapat persetujuan untuk dilanjutkan kepada tahap pengembangan. Dalam pengembangan, penugasan yang di-*assign* condong ke arah *backend*. Pada *backend*, *progress* yang ada berupa pembuatan fitur *login*, fitur *register*, serta sistem autentikasi dan lupa *password*.

Pada minggu ke-delapan, *backend website* dilanjutkan dengan penambahan fitur penambahan mesin, serta mesin mingguan, fitur pengiriman *email* otomatis.

Pada minggu ke-sembilan, *database* pada *website* sedikit disesuaikan dan menyelesaikan fitur-fitur lainnya dalam *website*.

Pada minggu ke-sepuluh hingga ke-tiga belas, *website* dilanjutkan dan direvisi sesuai dengan perubahan dan permintaan klien. Hal ini meliputi perpindahan fitur penambahan mesin ke dalam halaman *admin*, penambahan fitur *import* dan *export* untuk kebutuhan *backup* aplikasi, halaman dan *role* tambahan untuk gudang dengan sedikit fitur tambahan dalam halaman tersebut, dan perubahan sedikit pada fungsi yang dibuat karena kurang sesuai dengan kebutuhan.

Pada minggu ke-empat belas, klien melakukan perubahan drastis pada permintaan *website*. Untuk menyesuaikan *website* dengan *website* lainnya yang

sudah ada di dalam perusahaan, maka perlu dilakukan perubahan dari penggunaan *stack MERN* menjadi *Laravel*. Hal ini terjadi untuk menghindari perbedaan pada perusahaan yang dapat menjadi masalah yang cukup besar nantinya karena *website* harus dibuat ulang.

Pada minggu ke-lima belas hingga ke-tujuh belas, *website* mulai dibuat dan diganti menggunakan bahasa *Laravel*. Fitur serta tampilan yang ada pada *website* baru yang menggunakan *Laravel* sama persis dengan sebelumnya sehingga tidak ada perubahan signifikan yang terjadi.

3.4 Perencanaan

Pengerjaan *website* JPM dikerjakan dua kali yaitu dengan *stack MERN* dan *stack TALL*. Pada pembuatan *website* JPM, dilakukan perancangan dengan *sitemap*,*flowchart*, dan pembuatan *Entity Relation Diagram*

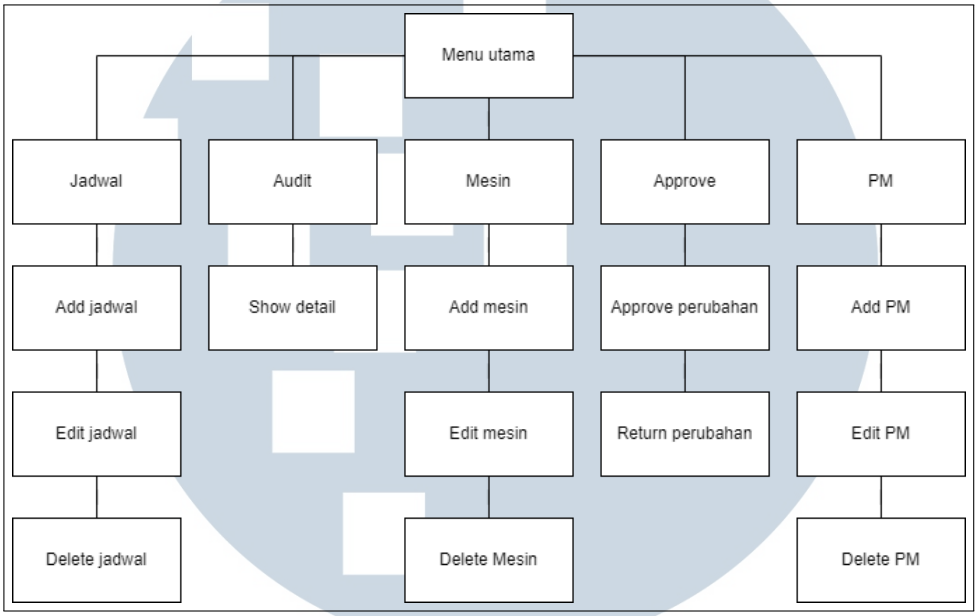
3.4.1 Teknologi yang digunakan

Dalam pengerjaan rancang bangun *website* JPM adapun beberapa teknologi yang digunakan sebagai sarana penunjang kegiatan pembuatan *website* yaitu:

- *figma*
- *Visual Studio Code*
- *Insomnia*
- *MongoDB*
- *Express.js*
- *React.js*
- *Node.js*
- *PostgreSQL*
- *Tailwind*
- *Alpine.js*
- *Laravel*
- *Livewire*

3.4.2 *Sitemap Website* JPM

Pengerjaan *website* JPM ini memiliki struktur yang dibuat pada awal pembuatan *website* JPM.

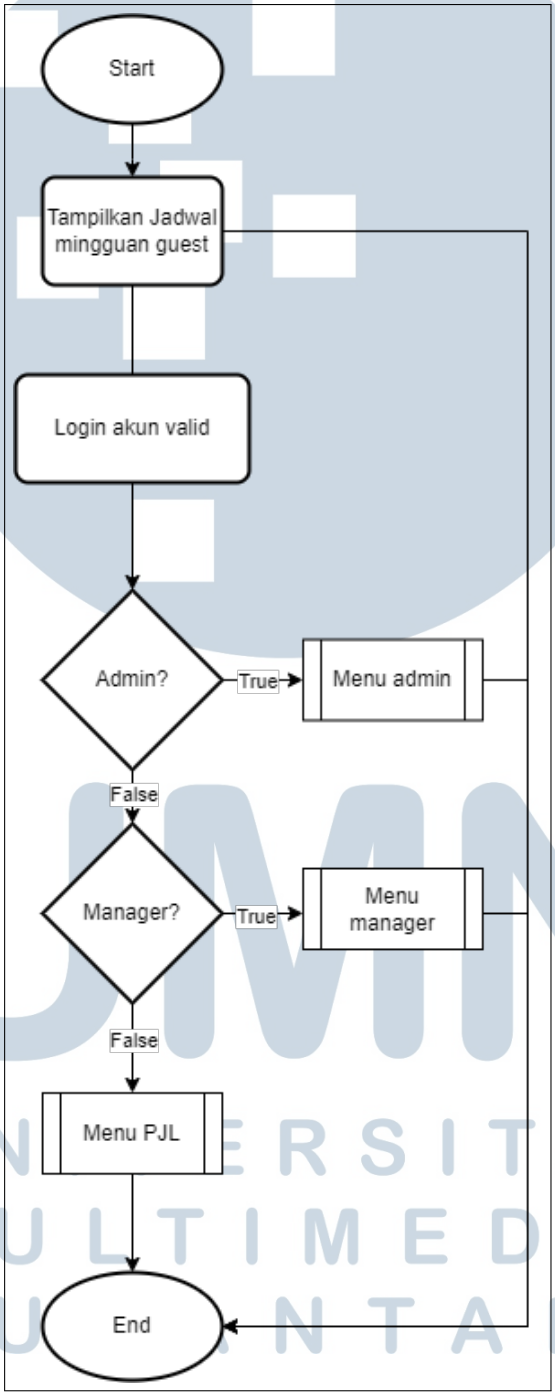


Gambar 3.1. *Sitemap Website* JPM

Gambar 3.1 merupakan *sitemap website* JPM yang berupa struktur utama dari *website*.*Website* JPM memiliki beberapa jenis akses yaitu *admin*, *manager* dan *PJL*. Pada halaman utama *PJL* terdapat beberapa fitur, yaitu penjadwalan, audit, dan *PM*. Pada halaman penjadwalan dan *PM*, pengguna dapat melakukan penambahan jadwal, penyuntingan jadwal dan penghapusan jadwal. Pada audit, pengguna dapat melihat detail audit yang ada. Pada akses *manager*, menu utama yang ditampilkan merupakan halaman *approval* yang berisikan fitur *approve* perubahan serta *return* perubahan. Terakhir, *admin* dapat menambah mesin, menyunting mesin, serta menghapus mesin.

3.4.3 *Flowchart Website* JPM

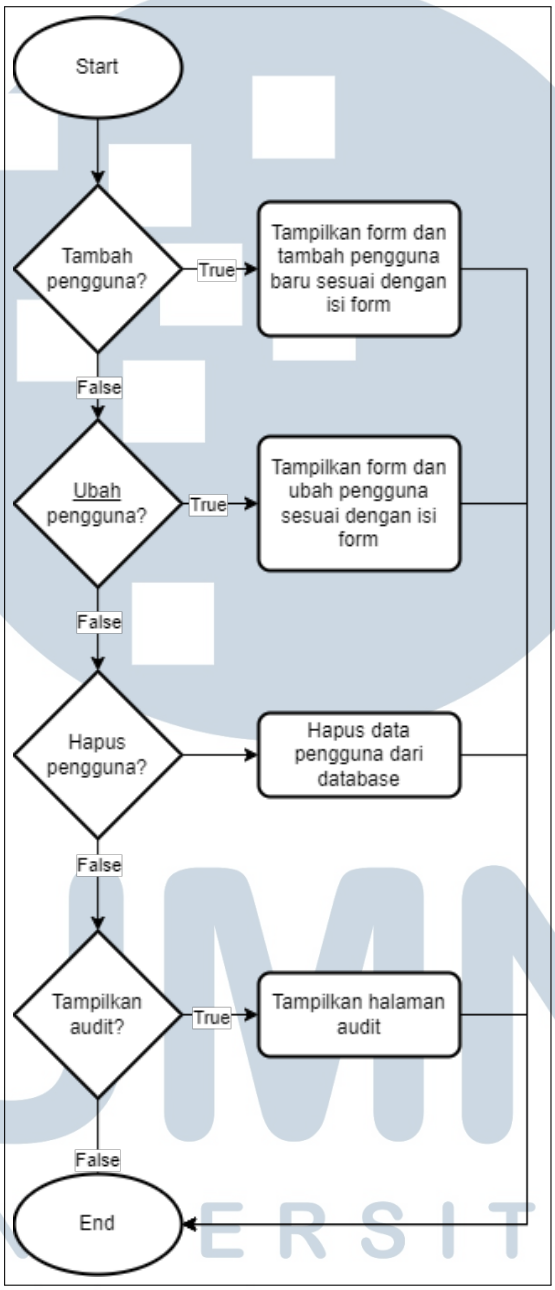
A. *Flowchart* Proses utama



Gambar 3.2. Gambar *flowchart* proses utama

Gambar 3.2 merupakan *flowchart* proses utama pada *website* JPM. *Flowchart* ini menampilkan beberapa akses yang ada pada *website*. Pada saat *website* dibuka, akan ditampilkan menu *guest*. Menu *guest* berisi jadwal mingguan yang telah dibuat oleh *PJL*. Jika pengguna memiliki akses *guest*, pengguna tidak dapat mengubah mesin mingguan, menambah mesin, atau kegiatan lainnya. Pengguna hanya bisa melihat jadwal serta melihat audit atau audit perubahan yang dilakukan pada mesin. Jika pengguna menekan tombol *login*, akan ditampilkan *form* untuk melakukan aksi *login*. Jika pengguna memasukkan akun yang *valid*, maka program akan mengecek *role* apakah yang dimiliki oleh pengguna tersebut apakah *admin*, *manager*, atau *PJL*. Setelah itu program akan mengalihkan pengguna ke halaman *role* mereka masing-masing.

B. *Flowchart* Menu *Admin*

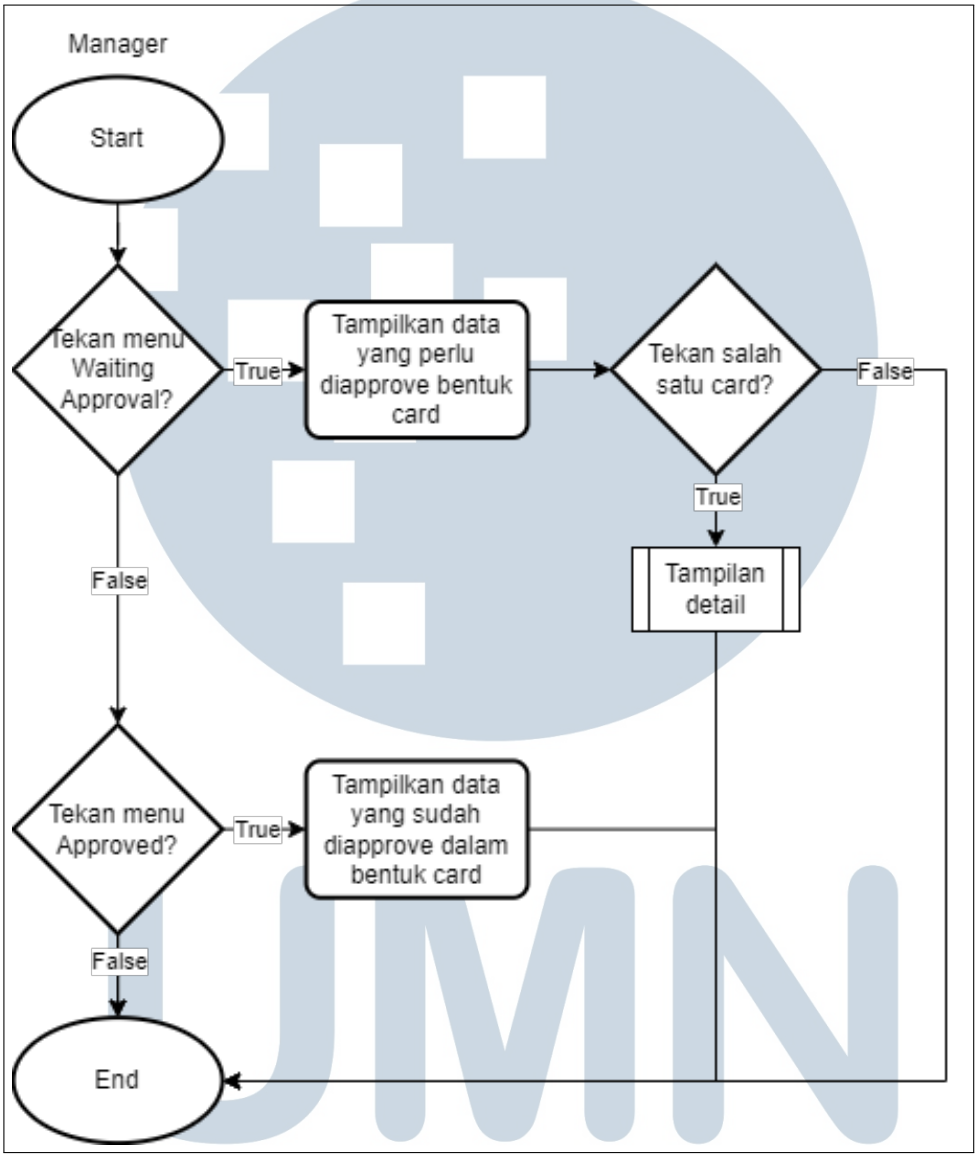


Gambar 3.3. Gambar *flowchart* menu *admin*

Pada halaman utama menu *admin*, akan ditampilkan semua data pengguna yang sudah terdaftar. Pada menu ini terdapat 4 hal yang dapat dilakukan oleh *admin* yaitu: membuat pengguna baru, mengubah pengguna, menghapus pengguna, dan melihat audit. Pada pembuatan pengguna baru, *admin* akan diberikan *form* untuk mengisi nama, *email*, *password*, dan *role* dari pengguna baru tersebut. Jika

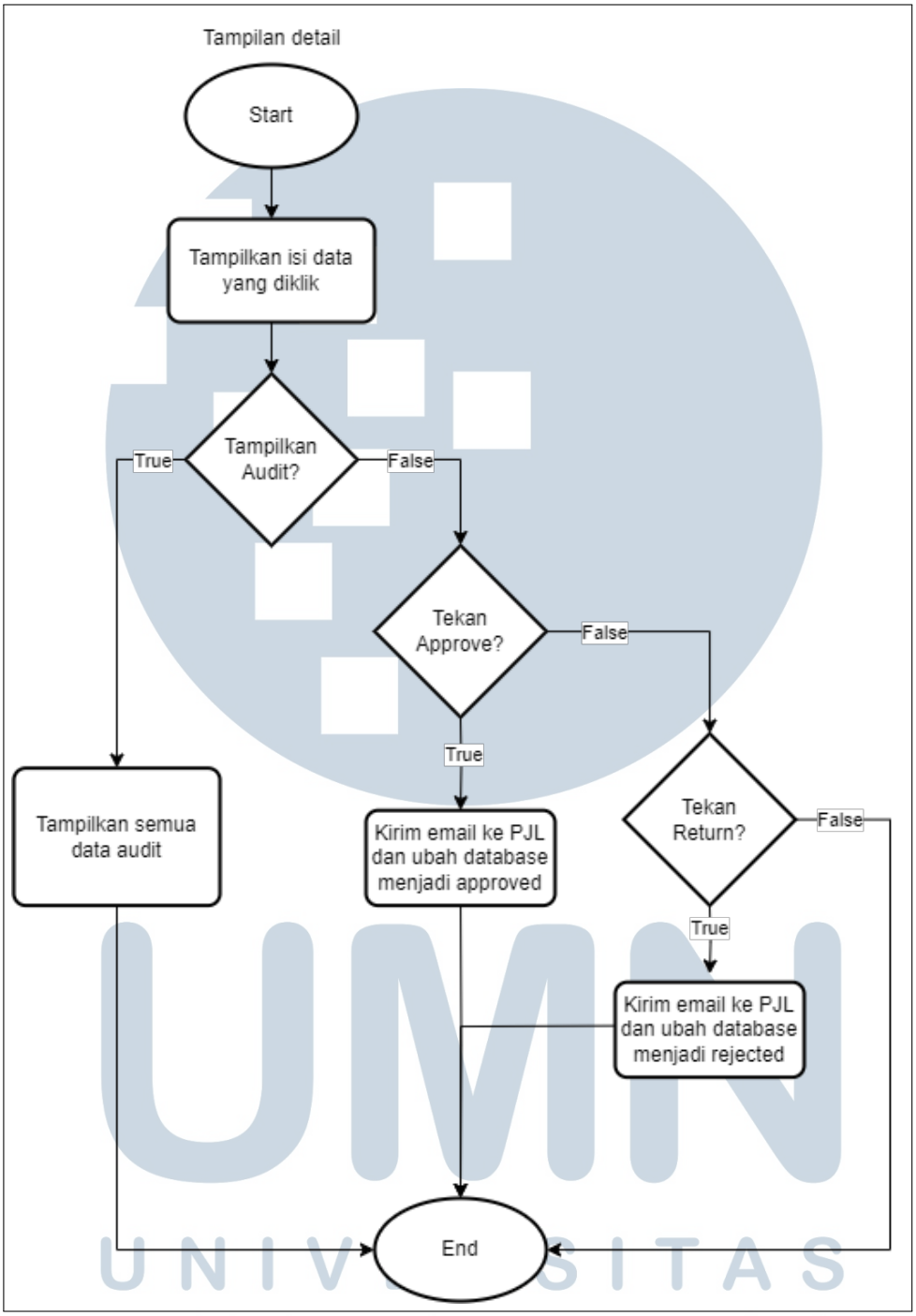
admin menekan *submit*, maka pengguna baru tersebut berhasil dibuat. Opsi kedua adalah mengubah pengguna, *form* yang ditampilkan pada pengubahan pengguna sama persis dengan *form* pembuatan pengguna. Beda utama dari *form* ini adalah, semuanya tidak wajib diisi. Jika *admin* tidak mengisi salah satu bagian dari *form*, maka program akan menganggap bagian tersebut tidak dirubah dan tetap menyimpan data sebelumnya. Opsi ketiga adalah menghapus pengguna. *admin* hanya perlu menekan tombol hapus pengguna dan program akan memberikan tampilan untuk konfirmasi kedua. Jika *admin* menekan tombol hapus 2 kali, maka data pengguna tersebut akan dihapus dari *database*. Opsi terakhir adalah audit. Jika *admin* menekan tombol ini, maka sistem akan menampilkan segala audit yang telah dilakukan oleh pengguna seperti penambahan mesin, penyuntingan mesin, penghapusan mesin dan lainnya.

C. *flowchart* Menu *Manager*



Gambar 3.4. Gambar *flowchart* menu *manager*

Pada halaman menu *manager* terdapat 2 pilihan yang dapat dilakukan oleh *manager* melihat daftar jadwal yang belum di-*approve* (*Waiting approval*) dan melihat daftar jadwal yang sudah di-*approve* (*approved*). Jika *manager* menekan tombol *approved* maka *website* akan menampilkan semua JPM yang telah disetujui oleh *manager*. JPM ini bisa akan hilang dari halaman *approved* ketika JPM diubah oleh *PJL*. Jika *manager* menekan tombol *waiting approval*, maka *manager* akan diperlihatkan data yang perlu di-*approve* beserta minggu serta bulannya dalam bentuk *card*.

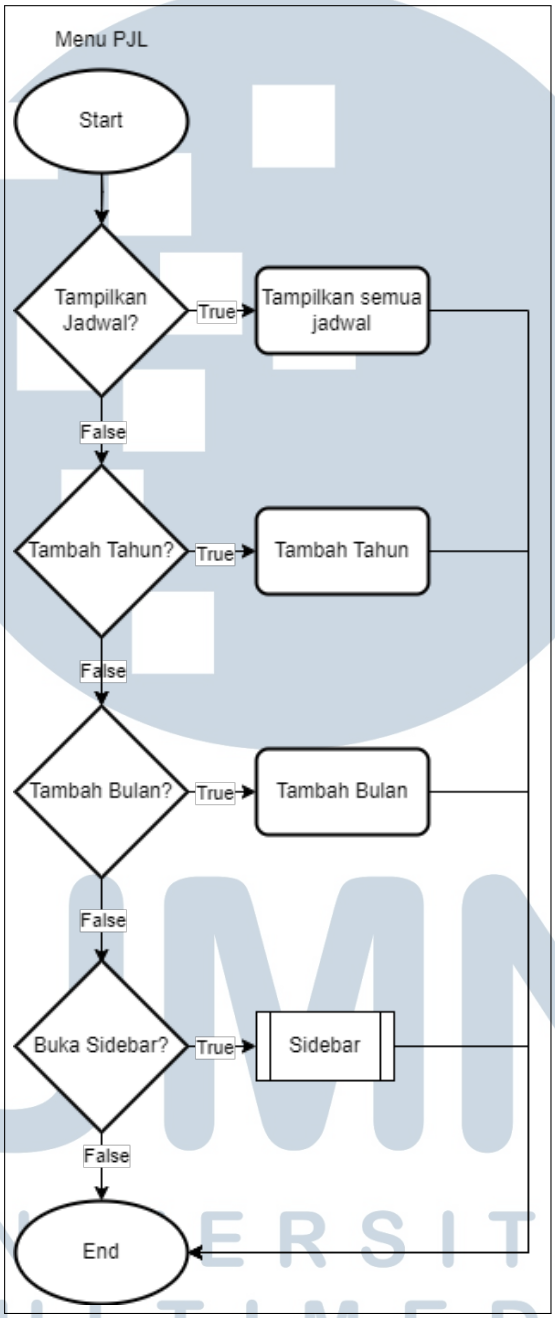


Gambar 3.5. Gambar *flowchart* tampilan detail *waiting approval*

Jika *manager* menekan salah satu *card* tersebut, maka akan ditampilkan detail dari perubahan yang dilakukan. Pada halaman detail, akan ada 3 pilihan tambahan yang dapat dipilih oleh *manager* yaitu audit, *approve*, dan *return*. Audit akan menunjukkan segala perubahan yang dilakukan pada data JPM tersebut sama seperti audit yang ada pada *guest* untuk memudahkan *manager* melihat

perubahan perubahan apa saja yang perlu dikonfirmasi. pilihan *Approve* akan mengembalikan status *Approve* pada data yang dikembalikan. Menekan tombol ini akan menghilangkan data tersebut dari tampilan *manager*. Setelah itu program akan mengirim *email* kepada *PJL* dengan *role line* yang sesuai dengan JPM yang baru saja di-*approve*. Fungsi *return* memiliki cara kerja yang sama dengan fungsi *approve* dengan perbedaannya berupa status yang dikembalikan akan *rejected* dan JPM yang di-*reject* tidak akan tampil di halaman *approved*.

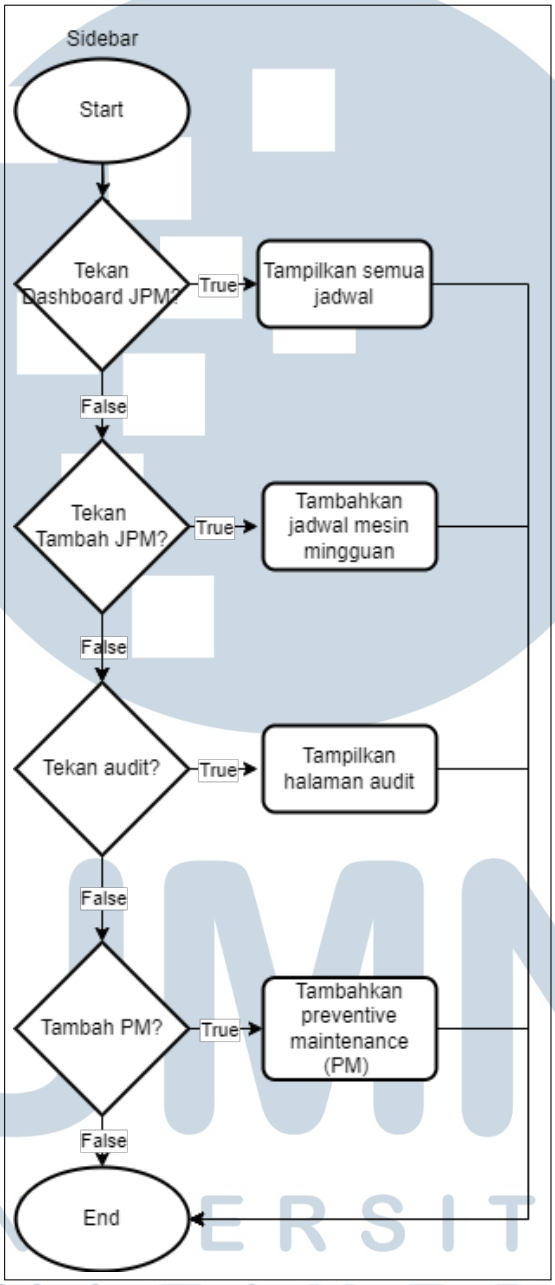
D. *flowchart* Menu *PJL*



Gambar 3.6. Gambar *flowchart* menu PJL

Pada *PJL*, akan ada 4 pilihan yaitu: tampilkan jadwal, tambah tahun, tambah bulan, dan buka *sidebar*. Pada tampilkan jadwal, pengguna akan ditampilkan semua jadwal. Pada tambah tahun, pengguna akan diminta untuk mengisi tahun dan men-*submit*. setelah di-*submit*, tahun akan muncul di sebelah tahun yang sebelumnya. Tambah bulan memiliki proses yang sama dengan tambah tahun,

tetapi sistem memproses tahun yang sekarang sedang dipilih terlebih dahulu baru membuat *modal* berisikan *form input* bulan.

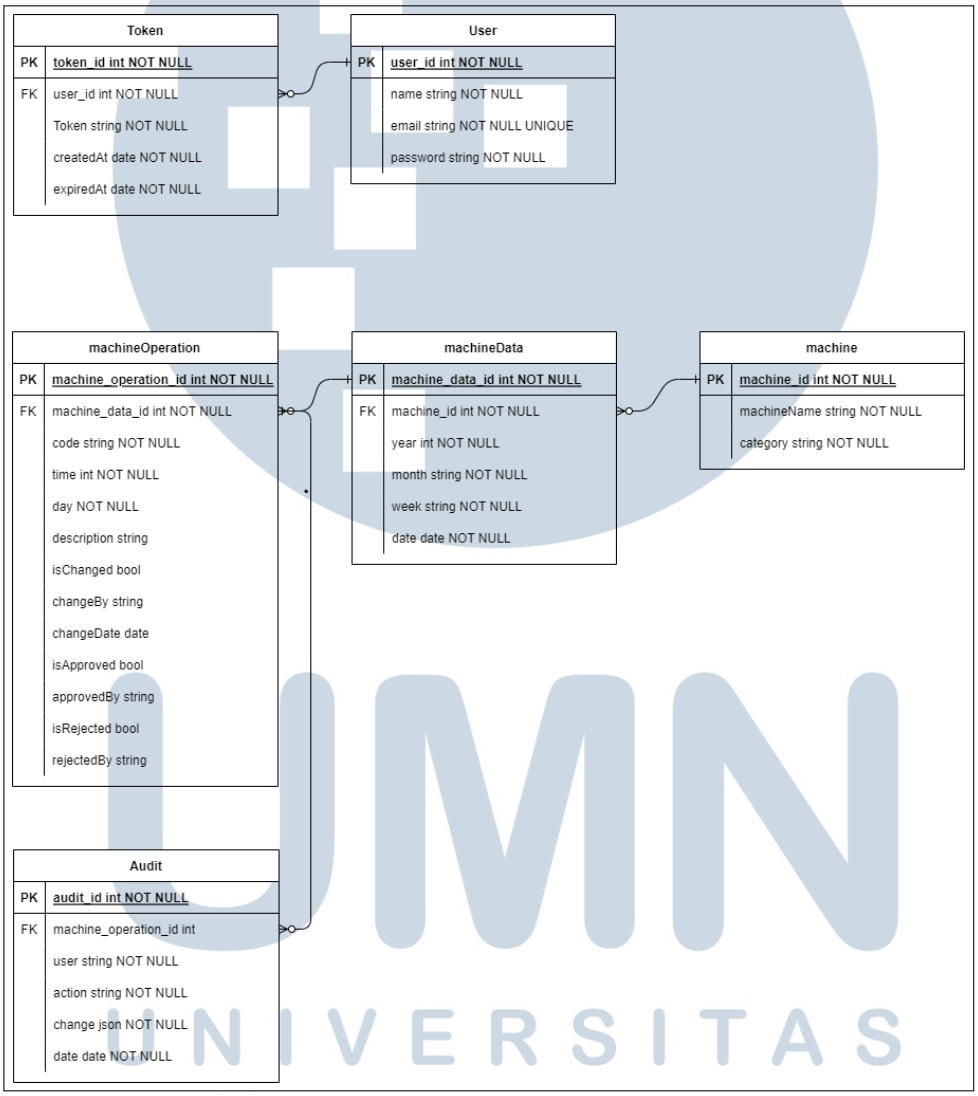


Gambar 3.7. Gambar *flowchart sidebar PJL*

Pada *sidebar* terdapat beberapa pilihan lagi yaitu: *dashboard* JPM, tambah JPM, audit, dan tambah JPM. Jika pengguna memilih opsi Pilih tahun *PJL*, maka akan ditampilkan bulan apa yang akan dipilih. Setelah pengguna memilih bulannya, maka sistem akan mengambil dari *database* data mingguan yang sesuai dengan tahun dan bulan yang dipilih. jika pengguna memilih untuk meng-*input* data mesin,

maka pengguna perlu menambahkan kode ruah, jam setelah keterangan untuk memasukkan data mesin. Jika pengguna meng-*hover* data maka deskripsi dari data tersebut akan ditunjukkan.

3.4.4 *Entity Relationship Diagram Website* JPM



Gambar 3.8. Gambar *Entity Relation Diagram* JPM

Pada *Database*, terdapat 6 tabel yang berisikan *Token*, *User*, *machineOperationDetail*, *machineData*, *machine*, dan audit.

A. Tabel *Token*

Pada tabel *Token* terdapat id untuk *token*, id dari *user* yang merupakan *foreign key* untuk mengetahui milik siapa *token* tersebut serta kapan *token* tersebut dibuat dan kapan *token* tersebut *expire*. Hal ini diperlukan untuk menghindari pengguna menggunakan 1 *token* yang sama untuk waktu yang tidak ditentukan lamanya.

B. Tabel *User*

Tabel *user* berisikan id dari *user* sendiri, nama *user* tersebut, *email*, dan *password*. semuanya berbentuk *string* kecuali id user. *email* dari *user* harus unik karena 1 *user* hanya boleh memiliki 1 *email*. dan juga sebaliknya, 1 *email* hanya boleh dipakai oleh 1 *user*.

C. Tabel *Machine*

Terpisah dari *user*, *Machine* merupakan *database* yang mengurus mesin mesin apa saja yang ada dalam sistem. Mesin ini bisa dipakai berkali-kali dan harus ditambahkan secara manual setiap minggunya. *database machine* terdiri dari id mesin, nama mesin dan kategori mesin dan semua tipe data-nya kecuali id berupa *string*. *database machine* ini ditunjuk oleh *database machineData*.

D. Tabel *machineData*

machineData merupakan tabel mesin mingguan yang perlu ditambah secara manual. Karena tabel ini bersifat mingguan, perlu ditambah data yang menunjukkan tanggal mesin ini berada secara akurat sehingga pada *machineData* terdapat id dari *machineData* itu sendiri, id dari *machine* yang berupa *foreign key*, *year* yang berupa *integer*, *month* yang berupa *string*, *week* yang berupa *string*, serta *date* yang berupa *date* sebagai tanggal penuhnya. Semua dari data ini tidak boleh kosong.

E. Tabel *machineOperation*

Tabel *machineOperation* menunjuk ke *machineData*. Tujuan dari *database* ini adalah sebagai *value* dari masing masing mesin pada mesin mingguan. Pada tabel ini terdapat id dari *machine operation* itu sendiri, id *foreign key* yang menunjuk

ke *machineData*, *code* yang merupakan kode ruah berbentuk *string*, *time* atau waktu pemakaian mesin yang berupa *int*, *day* atau hari pemakaian mesin, *description* atau deskripsi mesin yang merupakan *string*. Pada tabel ini juga perlu status untuk menunjukkan apakah data tersebut telah diubah, telah di-*approve*, atau telah di-*reject* sehingga ada *isChanged*, *changeBy*, *changeDate*, *isApproved*, *approvedBy*, *isRejected*, dan *rejectedBy*. *isChanged*, *isApproved*, dan *isRejected* berbentuk *boolean* yang bertujuan untuk mengecek apakah data tersebut diubah, di-*approve*, atau di-*reject*. *isChanged* akan menjadi *false* ketika data di-*approve* atau di-*reject*, dan demikian juga sebaliknya. *changeDate* akan ada ketika perubahan dilakukan. *changeDate* tidak akan hilang ketika data di-*approve* atau di-*reject*. *approvedBy* atau *rejectedBy* akan muncul sesuai dengan nama *manager* yang menyetujui atau mengembalikan perubahan.

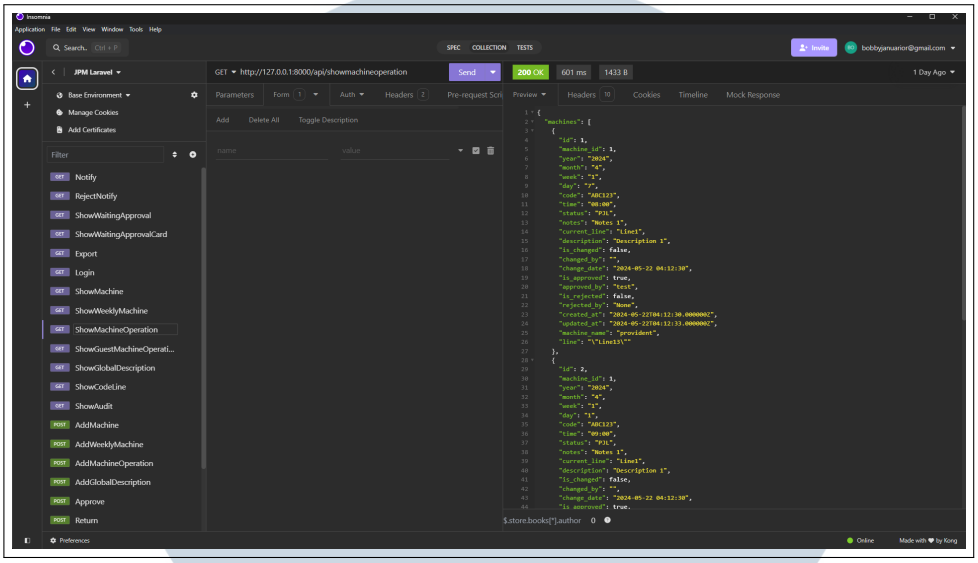
F. Tabel Audit

Tabel Audit menunjuk ke tabel*machineOperation*. Tujuan audit ini menunjuk ke *machineOperation* karena diperlukan id dari *machineOperation* supaya sistem mengetahui *machine operation* mana yang diubah. isi dari tabel ini yaitu id dari audit, id dari *machineOperation*, *user* yang berupa *string* dan diambil dari sesi, *action* berupa *string* yang merupakan aksi yang dilakukan berupa *add*, *edit*, atau *delete*. Selanjutnya ada *change* yang berupa *array* dalam *json* yang mengambil apa saja perubahan yang dilakukan, serta *date* untuk mengambil kapan perubahan tersebut terjadi.

3.5 Hasil Implementasi

Implementasi *backend* dari *website* diuji dengan menggunakan aplikasi *Insomnia*. Pada aplikasi *Insomnia*, *API* dari aplikasi bisa diuji tanpa harus menggunakan bantuan dari *frontend* sehingga dapat mempermudah pengerjaan *backend*

3.5.1 Hasil *API* Menggunakan Aplikasi *Insomnia*



Gambar 3.9. Gambar tampilan aplikasi *Insomnia* proyek *JPM*

Seperti pada gambar aplikasi *Insomnia* yang ada pada gambar 3.9, aplikasi *Insomnia* terdiri dari nama fungsi, *input*, dan *output*. Berikut merupakan semua fungsi yang telah dibuat dan *output*-nya pada aplikasi *Insomnia*:

Nama Fungsi	*Input*	*Output*
Notify	*NULL*	*"message": "Notifications sent successfully to all recipients."*
RejectNotify	*NULL*	*"message": "Notifications sent successfully to all recipients."*
Approve	*NULL*	*"message": "Approval successful"*
Return	*NULL*	*"message": "Return successful"*

ShowWaitingApproval	”*year*”: ”2024”, ”*month*”: ”4”, ”*week*” : ”1”	”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”, ”*day*”: ”1”, ”*code*”: ”ABC123”, ”*time*”: ”09:00”, ”*status*”: ”*PJL*”, ”*notes*”: ”*Notes 1*”, ”*currentline*”: ”*Line1*”
ShowWaiting ApprovalCard	*NULL*	”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”
ShowMachine	*NULL*	”id”: 1, ”*machinename*”: ”*Test*”, ”*category*”: ”*CategoryTest*”, ”*line*”: ”*Line1*”, ”*createdat*”: ”2024-05-08T01:32:44.000000Z”, ”*updatedat*”: ”2024-05-08T01:32:44.000000Z”
ShowWeeklyMachine	”*line*”: ”*Line1*”, ”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”,	”id”: 1, ”*machineid*”: 1, ”*machinename*”: ”*Machine* 1”, ”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”, ”*date*”: ”2024-04-01 00:00:00”, ”*createdat*”: ”2024-05-22T04:12:30.000000Z”, ”*updatedat*”: ”2024-05-22T04:12:30.000000Z”

ShowMachineOperation	”*line*”: ”*Line1*”, ”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”,	”id”: 1, ”*machineid*”: 1, ”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”, ”*day*”: ”7”, ”*code*”: ”ABC123”, ”*time*”: ”08:00”, ”*status*”: ”*PJL*”, ”*notes*”: ”*Notes 1*”, ”*currentline*”: ”*Line1*”, ”*description*”: ”*Description* 1”, ”*ischanged*”: *false*, ”*changedby*”: ””, ”*changedate*”: ”2024-05-22 04:12:30”, ”*isapproved*”: *true*, ”*approvedby*”: ”*test*”, ”*isrejected*”: *false*, ”*rejectedby*”: ”*None*” ”*createdat*”: ”2024-05-22T04:12:30.000000Z”, ”*updatedat*”: ”2024-05-22T04:12:33.000000Z”, ”*machinename*”: ”*provident*”, ”*line*”: ””*Line13*””

ShowGuest MachineOperation	”*line*”: ”*Line1*”, ”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”,	”id”: 1, ”*machineid*”: 1, ”*year*”: ”2024”, ”*month*”: ”4”, ”*week*”: ”1”, ”*day*”: ”7”, ”*code*”: ”ABC123”, ”*time*”: ”08:00”, ”*status*”: ”*PJL*”, ”*notes*”: ”*Notes 1*”, ”*currentline*”: ”*Line1*”, ”*description*”: ”*Description* 1”, ”*ischanged*”: *false*, ”*changedby*”: ””, ”*changedate*”: ”2024-05-22 04:12:30”, ”*isapproved*”: *true*, ”*approvedby*”: ”*test*”, ”*isrejected*”: *false*, ”*rejectedby*”: ”*None*” ”*createdat*”: ”2024-05-22T04:12:30.000000Z”, ”*updatedat*”: ”2024-05-22T04:12:33.000000Z”, ”*machinename*”: ”*provident*”, ”*line*”: ””*Line13*””
ShowGlobalDescription	*NULL*	”*globalDescription*”: ”*This is global description*”
ShowCodeLine	*NULL*	”*code*”: ”*ABC123*”, ”*line*”: ”*Line7*”

showAudit	*NULL*	”*auditid*”: 1, ”*machineoperationid*”: *null*, ”*action*”: ”*approve*”, ”*changes*”: ”*originalstate*”: ”*day*”: ”7”, ”*code*”: ”ABC123”, ”*time*”: ”08:00”, ”*description*”: ”*Description* 1”, ”*isChanged*”: *true*, ”*isApproved*”: *false*, ”*changedBy*”: ”*None*” , ”*newstate*”: ”*day*”: ”7”, ”*code*”: ”ABC123”, ”*time*”: ”08:00”, ”*description*”: ”*Description* 1”, ”*isChanged*”: *false*, ”*isApproved*”: *true*, ”*changedBy*”: ””
addMachine	*machineName* : ”*Test*” *category* : ”*CategoryTest*” *line* : ”*Line1*”	”*message*”: ”*Machine added successfully*”
AddWeeklyMachine	*machineName* : ”*qui*” *week* : ”3”	”*message*”: ”*Machine added successfully*”
addMachineOperation	*day* : ”1” *code* : ”*test*” *time* : ”07:00* *status* : ”*Supervise*” *notes* : ”*Lorem Ipsum Dolor Sit Amet*”	”*message*”: ”*Machine operation added successfully*”

addGlobalDescription	*description*: ”*Test*”	”*message*”: ”*Global description already exists*”
editMachineOperation	*day* : ”1” *code* : ”*test*” *time* : ”09:00* *status* : ”*Supervise*” *notes* : ”*Lorem Ipsum Dolor Sit Amet*”	”*message*”: ”*Machine operation edited successfully*”
deleteWeeklyMachine	*NULL*	”*message*”: ”*Weekly Machine deleted successfully*”
deleteMachineOperation	*NULL*	”*message”*: ”*Machine operation deleted successfully*”
deleteGlobalDescription	*NULL*	”*message”*: ”*Global description deleted successfully*”

A. Fungsi *Approval*

Pada fungsi *notify* dan *notifyRejection*, tidak ada *input* yang perlu dimasukkan karena fungsi tersebut berupa fungsi *GET* yang bertujuan untuk menotifikasi *PJL* terkait sesuai dengan *role email PJL* masing-masing. *Role email* pada *PJL* bisa lebih dari satu, contohnya *PJL* A memiliki *role email Line* 1 dan *Line* 2, maka fungsi *notify* ini akan mengirim *email* yang ditujukan kepada *Line* 1 dan *Line* 2 kepada *PJL* tersebut.

Fungsi *notify* ini akan mengirim *email* dengan mengecek semua data dari pengguna dan mencari *Line* yang sesuai lalu akan dikumpulkan menjadi 1 buah *array* yang berisikan nama-nama yang akan dikirimkan. Dengan metode ini, pengiriman *email* akan dikumpulkan menjadi 1 setiap prosesnya dibandingkan dengan cara lainnya yang mengirimkan *email* ke setiap orangnya. Fungsi ini dimasukkan ke dalam tombol yang sama dengan fungsi *approve* pada *manager* sehingga ketika *manager* meng-*approve* suatu perubahan, *email* akan sekaligus dikirim sebagai pengingat pada *PJL*.

Kekurangan fitur pengiriman ini terletak pada batasan 500 pengiriman *email* setiap harinya sehingga fitur hanya diimplementasikan pada *approval* dari *manager* dan tidak pada *PJL* kepada *manager* untuk menghindari habisnya kuota harian dan mengurangi kemungkinan terjadinya *error* karena hal tersebut.

Perbedaan antara fungsi *notify* dengan *notifyRejection* hanya terletak pada isi dari pesan yang dikirimkan kepada *PJL*. *Notify* akan mengirimkan pesan yang merupakan pemberitahuan untuk mengecek perubahan yang terjadi terhadap JPM sedangkan *notifyRejection* berisikan pemberitahuan untuk segera merevisi ulang perubahan yang baru dilakukan.

Pada fungsi *Approve* dan *Return*, tidak ada *input* yang perlu dimasukkan karena fungsi ini merupakan *response* dari klik yang dilakukan pada *button*. Pada fungsi *approve*, fungsi akan mengubah nilai dari *isApproved* menjadi *true* serta mengubah *isChanged* menjadi *false*. Hal ini bertujuan untuk menghilangkan data tersebut dari tampilan *manager* dan sebagai tanda bahwa perubahan telah diterima. *output* dari fungsi ini berupa pesan sukses. Mirip dengan fungsi *approve*, *return* akan mengembalikan data kepada *PJL* untuk direvisi lebih lanjut. Fungsi ini akan mengubah *isRejected* menjadi *true* dan *isChanged* menjadi*false*.

B. Fungsi *Show*

Pada *website JPM* butuh banyak fungsi untuk menampilkan data-data yang ada pada *database*, oleh karena itu dibuat fungsi untuk memanggil semua data ini dengan fungsi *show* yang menggunakan *request GET*. Fungsi *show* ini terdiri dari *ShowWaitingApproval*, *ShowWaitingApprovalCard*, *ShowMachine*, *ShowWeeklyMachine*, *ShowMachineOperation*, *ShowGuestMachineOperation*, *ShowGlobalDescription*, *ShowCodeLine*, dan *ShowAudit*.

ShowWaitingApproval berfungsi sebagai fungsi yang akan menunjukkan jadwal yang baru saja direvisi oleh *PJL*. Jadwal ini nantinya akan memiliki tampilan yang sama persis seperti tampilan yang ada pada *Guest* atau *PJL*. Karena fungsi ini merupakan fungsi untuk memanggil data berdasarkan data yang dipilih dari data *card* sebelumnya, *input* yang perlu dimasukkan berupa tahun, bulan, serta minggu dari data yang dipilih. *output* yang dikeluarkan berupa tahun, bulan, minggu, hari, kode, waktu, *status*, *notes*, dan *currentline* atau *line* dari mesin tersebut. Semua data ini merupakan data-data yang ada dari *database machineOperation*.

ShowWaitingApprovalCard Memiliki fungsi yang berhubungan dengan *ShowWaitingApproval*, Fungsi ini merupakan fungsi yang akan dimasukkan ke

dalam *card* opsi yang bisa dipilih oleh *manager*. *Card* ini akan ditampilkan pada jadwal yang baru direvisi oleh *PJL*. Hal tersebut dapat diambil dari nilai *isApproved* yang akan berubah menjadi *false* ketika terjadi perubahan. Fungsi ini tidak membutuhkan *input* apapun dan akan mengeluarkan *output* tahun, bulan, dan minggu.

Fungsi *ShowMachine* merupakan fungsi yang bertujuan untuk menunjukkan semua mesin yang telah dibuat oleh *admin*. Fungsi ini nantinya akan berhubungan dengan fungsi *ShowWeeklyMachine* di mana hanya mesin dengan *Line* yang sama dengan *currentline* yang dapat ditambahkan ke dalam *Weekly Machine*. Fungsi ini tidak membutuhkan *input* apa pun dan akan mengeluarkan data dari *database Machine* yang berupa *id*, nama mesin, kategori mesin, *line* mesin, *createdat*, dan *updatedat*.

Selanjutnya fungsi *ShowWeeklyMachine* akan menampilkan mesin yang sudah ditambahkan pada minggu tersebut. Berbeda dengan *ShowMachine*, fungsi ini akan menampilkan mesin dari minggu tersebut dan bukan secara keseluruhan. Setiap minggunya mesin yang ada bisa berbeda sehingga fungsi ini dibuat untuk melihat data minggunya serta dibuat *database* khusus untuk fungsi ini. Fungsi ini membutuhkan *input* berupa *line*, tahun, bulan, dan minggu. *Output* dari fungsi ini berupa *id*, *machineId* yang merupakan *id* yang bisa didapat dari *database machine* terkait, nama mesin, tahun, bulan, minggu, tanggal, *createdat*, dan *updatedat*.

Fungsi *ShowMachineOperation* merupakan ekstensi dari fungsi *ShowWeeklyMachine* di mana fungsi ini berfungsi untuk menunjukkan operasi-operasi yang ada pada mesin minggu itu. Fungsi ini akan menyesuaikan *machineid* yang merupakan id dari *database MachineData* dengan mesin mingguan yang terkait sehingga hasil akhir akan berupa tabel *child* dari mesin mingguan. *Input* yang dibutuhkan pada fungsi ini berupa *line*, tahun, bulan, dan minggu sedangkan *output* yang berupa seluruh data dari *machineOperation* yaitu, *id*, *machineid* yang merupakan *id* dari *machineData*, tahun, bulan, minggu, hari, kode, waktu, *status*, *notes*, *currentline*, deskripsi, *isChanged*, *changedby*, *changedate*, *isApproved*, *approvedby*, *isRejected*, *rejectedby*, *createdAt*, *updatedAt*, *machineName*, dan *line*. *machineName* dan *line* berasal dari *database machine* yang ditunjuk oleh *machineData*.

Mirip dengan *ShowMachineOperation*, *ShowGuestMachineOperation* merupakan fungsi yang berfungsi untuk menunjukkan operasi-operasi yang ada pada mesin minggu itu. Pada *ShowGuestMachineOperation*, data yang ditunjukkan hanya data yang memiliki nilai *isApproved true* sehingga data-data yang belum di-

approve oleh *manager* tidak akan tampil. Data ini akan ditampilkan pada halaman *guest*. *Input* yang dibutuhkan pada fungsi ini sama dengan *ShowMachineOperation* yaitu *line*, tahun, bulan, dan minggu sedangkan *output* yang berupa seluruh data dari *machineOperation* yaitu, *id*, *machineid* yang merupakan *id* dari *machineData*, tahun, bulan, minggu, hari, kode, waktu, *status*, *notes*, *currentline*, deskripsi, *ischanged*, *changedby*, *changedate*, *isapproved*, *approvedby*, *isrejected*, *rejectedby*, *createdat*, *updatedat*, *machinename*, dan *line* dengan *machineName* dan *line* berasal dari *database machine* yang ditunjuk oleh *machineData*.

ShowGlobalDescription merupakan fungsi yang berguna untuk menampilkan semua deskripsi mingguan yang ditulis oleh *PJL*. *GlobalDescription* ini nantinya akan terletak di bawah tampilan JPM. Fungsi ini tidak memerlukan *input* dan akan mengeluarkan *output* "*globalDescription*" yang merupakan isi dari deskripsi tersebut.

Fungsi *ShowCodeLine* adalah fungsi yang berguna untuk menunjukkan semua kode dan *line* dari mesin. Fungsi ini dibutuhkan karena data kode dan data *line* berada di *database* yang berbeda serta tidak berhubungan langsung. Tidak diperlukan *input* untuk fungsi ini, dan fungsi ini akan mengeluarkan kode dan *line*.

Pada fungsi *showAudit* akan ditampilkan semua perubahan yang telah dibuat oleh pengguna dari semua akses kecuali *admin*. Audit memiliki salah satu kolom bernama *changes* yang berupa *json* penyimpan data lama dan data baru. kolom *changes* akan ditampilkan seluruhnya sehingga pengguna dapat mengetahui perbedaan dan perubahan yang dilakukan. Sama seperti beberapa fungsi *show* lainnya, *showAudit* tidak memerlukan *input* untuk dipanggil. *Output* dari fungsi ini berupa *auditid*, *machineoperationid* yang merupakan id dari *machineOperation*, *action* yang merupakan *string* berisikan tipe perubahan seperti *add*, *delete*, *edit*, atau*approve*, dan *changes* atau perubahan yang dilakukan. Pada kolom *changes*, akan berisi sesuai dengan data baru yang ada. Bila data yang diubah berupa *machineOperation* maka kolom *changes* ini akan menampilkan *MachineOperation*.

C. Fungsi *Add*

Fungsi *add* berfungsi sebagai fungsi penambah data ke dalam *database*. Ada 4 jenis fungsi *add* pada *website* JPM, 3 untuk mesin, mesin mingguan, dan operasi mesin dan 1 untuk *globalDescription*. Untuk cara kerja semua fungsi ini sama persis yaitu dengan menggunakan *HTTP Request POST* untuk menambahkan data ke dalam *database* masing-masing fungsi.

Fungsi *addMachine* digunakan untuk menambahkan mesin baru ke dalam *database*. Mesin ini nantinya akan bisa digunakan saat memilih mesin mingguan sesuai dengan *line* dari mesin. Fungsi ini hanya digunakan pada akses *admin* sehingga pengguna selain *admin* tidak bisa menyalahgunakan fungsi ini untuk menambahkan mesin fiktif. *Input* dari fungsi ini berupa nama mesin, kategori mesin, serta *line* dari mesin. Fungsi ini akan mengeluarkan *output* pesan sukses *Machine added successfully*.

Mirip dengan fungsi *addMachine*, fungsi *addWeeklyMachine* juga berfungsi untuk menambahkan mesin. Namun, fungsi ini hanya akan menambah mesin sebagai bentuk mesin mingguan. Data mesin mingguan yang dapat ditambahkan didapatkan dari mesin dengan *line* yang sama dengan *line* yang sekarang sedang dibuka. *Input* dari fungsi ini berupa nama mesin dan minggu mesin. Jika pengguna mencoba menambahkan mesin mingguan yang sama pada minggu, bulan, dan tahun yang sama, maka akan muncul pesan *error* untuk memberitahu pengguna bahwa mesin sudah ada. *Output* dari fungsi ini berupa pesan sukses *Machine added successfully*.

Fungsi *addMachineOperation* berguna untuk menambahkan operasi baru terhadap mesin mingguan yang ada. Sama dengan mesin mingguan, operasi yang ditambahkan hanya akan terkait dengan minggu tersebut dan tidak akan ada pada minggu yang berbeda. Jika pengguna mencoba untuk menambahkan operasi pada waktu yang sudah dipakai, maka pengguna akan mendapatkan pesan *error* karena adanya kesamaan jadwal. *Input* dari fungsi ini berupa hari, kode, waktu, *status*, dan *notes*. *Output* dari fungsi ini berupa pesan sukses *Machine operation added successfully*.

Terakhir terdapat *addGlobalDescription* yang berfungsi sebagai penambah deskripsi mingguan yang terdapat di bawah JPM. *Input* dari fungsi ini berupa deskripsi yang ingin ditambahkan oleh pengguna. Jika deskripsi yang sama sudah ada pada minggu, bulan, dan tahun yang sama, maka akan keluar *error* untuk memberitahu pengguna bahwa deskripsi tersebut sudah ada. *Output* dari fungsi ini adalah pesan sukses *Global description added successfully*.

D. Fungsi Edit

Fungsi *edit* merupakan fungsi yang berguna untuk mengubah data pada *database*. Fungsi ini menggunakan *HTTP Request PUT* untuk meng-*update* suatu data yang sudah ada pada *database*. Hanya terdapat 1 fungsi *edit* pada *website* JPM

ini yaitu *editMachineOperation*. Hal ini untuk menjaga integritas data sehingga data tidak mudah hilang. Ketika pengguna mengubah data pada mesin mingguan, maka semua prosesnya akan ikut terubah dan ini merupakan hal yang sulit untuk dicatat pada audit.

Fungsi *editMachineOperation* ini sama persis dengan fungsi penambahan operasi pada mesin (*addMachineOperation*), tetapi tidak perlu menghapus operasi. Jika pengguna mencoba untuk mengubah operasi ke waktu yang sudah dipakai, maka pengguna akan mendapatkan pesan *error* karena adanya kesamaan jadwal. *Input* dari fungsi ini berupa hari, kode, waktu, *status*, dan *notes*. *Output* dari fungsi ini berupa pesan sukses *Machine operation edited successfully*.

E. Fungsi *Delete*

Fungsi *delete* merupakan fungsi yang berguna untuk menghapus data yang sudah ada dalam *database*. Fungsi ini menggunakan *request DELETE* untuk menghapus data yang sudah ada pada *database*. Fungsi *delete* terdapat pada mesin mingguan, operasi mesin, dan deskripsi mingguan.

Fungsi *deleteWeeklyMachine* merupakan fungsi yang berguna untuk menghapus data mesin mingguan. Sistem penghapusan ini hanya bisa dilakukan ketika tidak ada operasi mesin yang berhubungan langsung dengan mesin mingguan ini. Tidak ada *input* yang dibutuhkan dan *output* yang dikeluarkan berupa pesan sukses *Weekly Machine Deleted Successfully*.

Fungsi *deleteMachineOperation* sama dengan fungsi *deleteWeeklyMachine* yang hanya berfungsi untuk menghapus data yang sudah ada pada operasi mesin di mesin mingguan tersebut. Namun, penghapusan operasi ini tidak memiliki batasan sehingga operasi bisa dihapus tanpa ada *requirement* yang perlu dipenuhi layaknya pada mesin mingguan. Tidak ada *input* yang dibutuhkan dan *output* yang dikeluarkan berupa pesan sukses *Weekly Machine Deleted Successfully*.

Terakhir fungsi *deleteGlobalDescription* berfungsi untuk menghapus deskripsi dari *database*. Tidak ada *input* yang dibutuhkan dan *output* yang dikeluarkan berupa pesan sukses *Weekly Machine Deleted Successfully*.

3.6 Kendala dan Solusi yang Ditemukan

3.6.1 Kendala

Salah satu kendala yang ditemui ketika membuat *website* JPM ini adalah ketidakpastian keputusan dari klien yang membutuhkan. Jika hari ini ditentukan akan mengerjakan suatu fitur, belum tentu fitur yang sama akan tetap sama pada minggu depan. Banyaknya perubahan menyebabkan mundurnya jadwal pengerjaan dan *deadline* pengerjaan yang diberikan. Selain itu adanya perubahan bahasa yang perlu digunakan. Rencana awal dari *website* JPM akan dibuat dengan menggunakan *stack MERN*, tetapi karena adanya konflik antara klien, atasan, dan tim *IT*, diputuskan bahwa ada pergantian dari *MERN* menjadi *Laravel*. Hal ini menyebabkan mundurnya segala *timeline* dan hilangnya semua *progress* yang sudah dibuat saat menggunakan *MERN*. Hilangnya *progress* ini membuat program baru menggunakan *Laravel* belum jadi sepenuhnya pada bagian *frontend*.

Selain ketidakpastian, penambahan fitur yang di-*request* oleh klien terhitung sangat banyak. Hal ini menyebabkan banyaknya perubahan pada *figma* dan banyak revisi yang perlu dilakukan sebelum implementasi *website*. Penambahan fitur ini juga terjadi ketika *website* sedang dibuat. Hal ini menyebabkan banyak perubahan pada struktur *database*, struktur dari fungsi *backend website*, atau tampilan dari *website*.

3.6.2 Solusi

Solusi dari ketidakpastian klien adalah dengan memberikan beberapa opsi dari awal. Opsi-opsi ini bisa digunakan sebagai ”ide” untuk klien sehingga mengurangi kesempatan klien untuk mendapatkan ”ide” baru yang nantinya akan menjadi perubahan pada *website* saat dibuat.

Kedua, penambahan fitur yang banyak memiliki solusi yang minim. Hal ini terjadi karena banyaknya aturan dan *SOP* yang harus dipatuhi oleh perusahaan. Salah satu solusi untuk penambahan fitur ini yaitu dengan cara membuat suatu fitur yang dapat menjalankan tugas tugas baru tersebut dengan fungsi yang sama persis. Halaman yang dibuat oleh bagian *frontend* juga menjadi sama persis dengan halaman lainnya untuk beberapa fitur baru seperti fitur gudang yang memiliki akses yang dibilang mirip dengan akses ”*guest*”.