BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kegiatan magang dilaksanakan pada posisi SCADA *engineer* dalam perusahaan, dengan tugas berupa mempelajari tentang sistem dan pekerjaan yang sedang berjalan di dalam perusahaan dan membantu *engineer* senior dalam operasi sistem dan pekerjaan tersebut. Selain membantu dalam operasi yang telah berjalan, dilakukan juga pembelajaran mengenai proyek berupa perancangan HMI untuk proyek baru. Pelaksanaan kedua pengerjaan tersebut diawasi oleh seorang supervisor, yang berupa staf dengan posisi *engineer* senior dalam perusahaan.

3.2 Tugas dan Uraian Kerja Magang

Tugas utama yang dikerjakan dalam proses kerja magang adalah pembuatan HMI dan sistem notifikasi untuk memantau mesin *wood chipper* pada situs yang berlokasi jauh. Mesin *wood* chipper yang digunakan adalah mesin *custom* yang berdasarkan mesin *microwoodchipper* Bruks WH500. Mesin tersebut dapat dilihat pada Gambar 3.1.

Gambar 3.1 Mesin *wood chipper*

Berdasarkan spesifikasi dari klien, variabel yang ingin diamati pada mesin tersebut adalah:

- Sensor arus listrik, yang memberikan ukuran arus listrik yang mengalir dalam mesin. Besar arus akan digunakan untuk mendeteksi bila mesin sedang dalam kondisi menyala atau mati.
- Sensor Temperatur, yang berfungsi untuk mengukur temperatur kerja mesin, sehingga dapat mencegah terjadinya overheating.
- Sensor Rotasi Motor, yang mengukur kecepatan rotasi dari baling-baling pemotong yang digunakan di dalam mesin. Terdapat dua buah sensor rotasi motor karena digunakan dua buah baling-baling pada mesin.
- Detektor Logam digunakan untuk mendeteksi masuknya objek yang tidak diinginkan ke dalam mesin, yang dapat menyebabkan kerusakan.

Variabel-variabel yang diukur dan besarannya pada kondisi operasi normal dapat dilihat pada tabel 1.

Tabel 3.1 Spesifikasi Mesin Wood Chipper

Variabel yang diukur	**Nilai pada operasi normal**
Arus listrik	500A
Temperatur Mesin	150-180°C
Kecepatan putaran motor	2400-3600 rpm
Detektor Logam	Tidak aktif

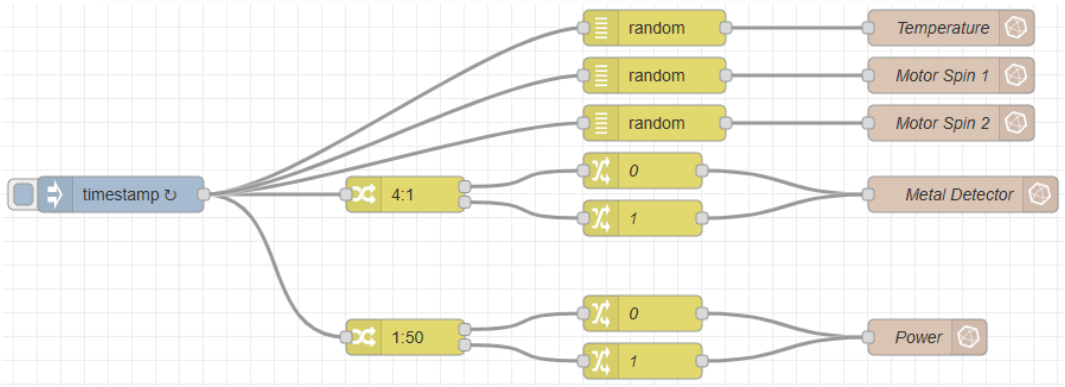
Untuk memenuhi kebutuhan yaitu pengamatan dari jarak jauh, maka seluruh sistem akan diunggah ke jaringan, agar dapat diakses dengan internet. Data yang diterima oleh sensor di tempat akan diunggah ke *database* yang berada dalam jaringan. Tampilan HMI yang dibuat akan dapat mengakses data yang tersedia pada *database* tersebut untuk ditampilkan.

Berdasarkan kebutuhan-kebutuhan tersebut, maka digunakan beberapa *software* untuk pembuatan HMI, yaitu Node-RED, InfluxDB, dan Grafana. Fungsi dari *software-software* tersebut adalah:

- InfluxDB merupakan *software time series database* (TSDB), yaitu *database* yang digunakan untuk menyimpan data yang berupa *time series* untuk keperluan *monitoring* atau pemrosesan data *real time*. Dalam proyek ini InfluxDB digunakan sebagai *database* untuk menyimpan data mentah dari sensor-sensor yang digunakan dalam mesin sebelum divisualisasikan pada HMI yang dibuat.
- Node-RED merupakan bahasa pemrograman berbasis *flow* yang dapat dijalankan dalam *web browser*. Dalam proyek ini Node-RED digunakan untuk membuat simulasi dari data yang akan diakuisisi dari mesin sebenarnya.
- Grafana adalah *software* visualisasi data yang dapat menampilkan grafik tabel, dan peringatan kepada pengguna. Grafana digunakan sebagai *software* utama untuk pembuatan HMI.

3.2.1 Simulasi Mesin dengan Node-RED

Untuk menyimulasikan kondisi mesin yang diterima oleh sensor, digunakan program Node-RED seperti pada gambar 3.2



Gambar 3.2 Program Node-RED untuk menyimulasikan kondisi mesin

Program yang digunakan terdiri dari *node-node* yang memiliki fungsi sendiri, antara lain:

- Node *inject*

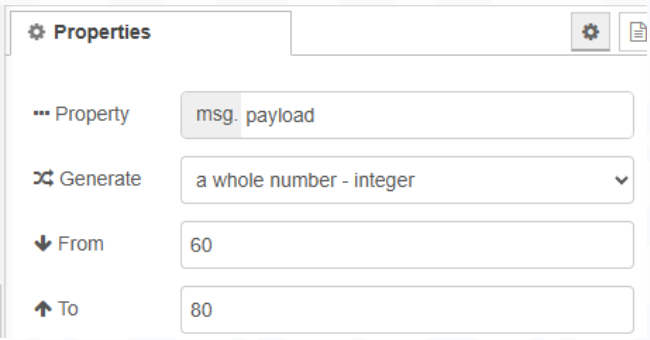
Gambar 3.3 node *inject*

Node *inject* seperti pada gambar 3.3 berfungsi sebagai awalan program, yang memasukkan sebuah data atau sinyal untuk diteruskan ke *node-node* berikutnya. Dalam Program 3.2, node inject digunakan untuk mengirimkan sinyal setiap interval 1 detik. Sinyal ini memiliki isi data atau *payload* yang berisi data waktu saat sinyal tersebut terkirim.

- Node *random*

Gambar 3.4 node random

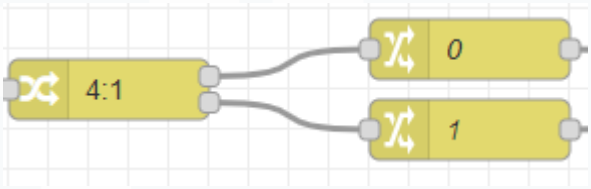
Node *random* seperti pada gambar 3.4 akan mengubah *payload* dari sinyal menjadi nilai angka acak di dalam *range* yang dapat ditentukan. Dalam program 3.2 node *random* digunakan untuk menyimulasikan nilai yang dihasilkan oleh sensor pada mesin. Sebagai contoh, pada node random untuk menyimulasikan nilai temperatur digunakan *range* nilai yang serupa dengan spesifikasi kerja mesin pada Tabel 3.1, seperti pada Gambar 3.5.



Gambar 3.5 pengaturan pembuatan data temperatur mesin

- Node *random output* dan node *change*

  Node *random output* akan meneruskan sinyal yang masuk ke node tersebut ke salah satu *output* dari node tersebut secara acak dengan *weight* yang dapat diatur. Node *change* berfungsi untuk mengubah *payload* dari sinyal menjadi data yang ditentukan, sehingga ketika dikombinasikan seperti pada Gambar 3.6 dapat dibuat fungsi untuk membuat output *boolean* dengan probabilitas berbeda.



Gambar 3.6 node random output dan *change*

- Node influxdb out

  Node influxdb out berfungsi untuk meneruskan *payload* sinyal ke influxdb. .

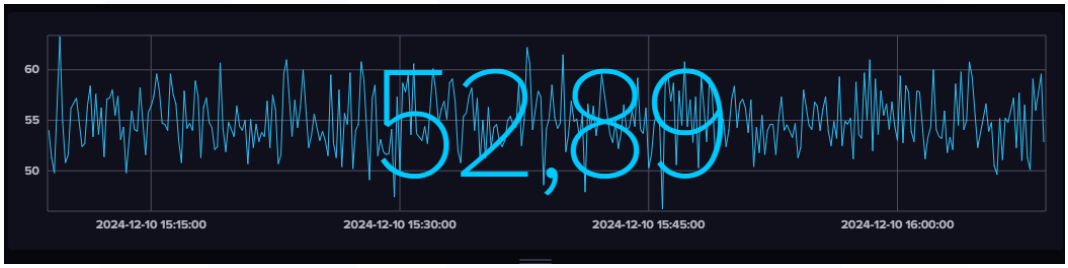
Gambar 3.7 Node influxdb out

Terdapat lima buah variabel yang disimulasikan, yaitu nilai temperatur, 2 buah nilai kecepatan putaran motor, sebuah sensor pendeteksi logam, dan sebuah sensor arus listrik yang berfungsi sebagai indikasi bahwa mesin sedang berjalan atau berhenti. Nilai variabel temperatur dan kecepatan motor berupa bilangan bulat. Nilai dari sensor pendeteksi logam berupa nilai *boolean* yaitu hanya 1 dan 0, dan nilai dari sensor arus juga direduksi menjadi nilai *boolean* saja.

3.2.2 Penyimpanan Data Sensor ke Database

Data yang dibuat dengan menggunakan Node-RED atau didapatkan oleh sensor pada mesin akan disimpan ke *database* influxdb. Pada simulasi yang digunakan, data dikirimkan ke influxdb dengan menggunakan node ‘influxdb out’ pada algoritma Node-RED, seperti pada Gambar 3.7.

Data yang disimpan pada influxdb akan disimpan terus selama durasi tertentu. Durasi tersebut dapat diubah sesuai dengan kebutuhan. Data tersebut juga dapat ditampilkan dalam bentuk grafik seperti pada Gambar 3.8.

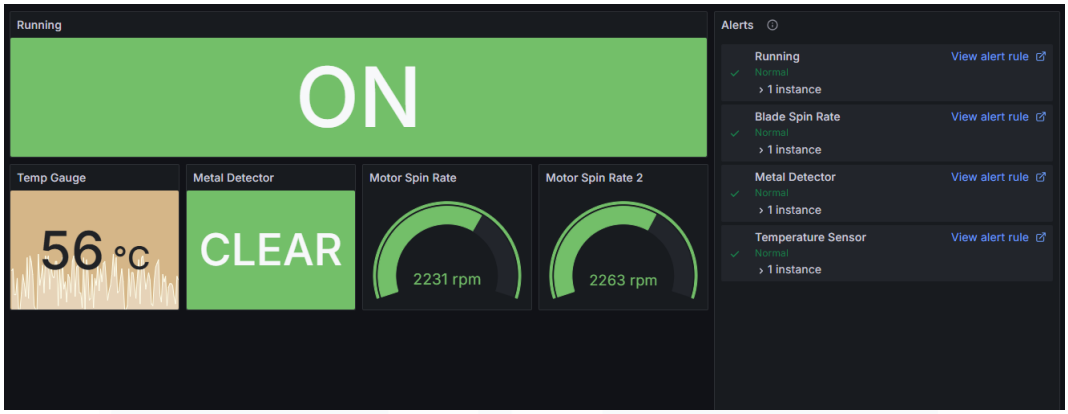


Gambar 3.8 Tampilan data pada influxDB

3.2.3 Penyajian Data dengan Grafana

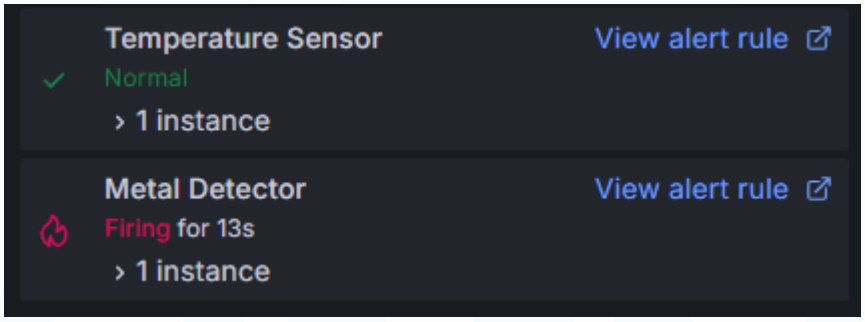
Data yang tersimpan di database influxDB akan ditarik dan ditampilkan dengan menggunakan aplikasi Grafana. Data tersebut dapat ditampilkan ke Grafana dengan berbagai jenis visualisasi.

Desain HMI yang digunakan adalah indikator untuk kecepatan motor, grafik untuk temperatur, dan *single stat* yang menunjukkan kondisi keaktifan mesin dan juga detektor logam. Desain ini dapat dilihat pada Gambar 3.9.



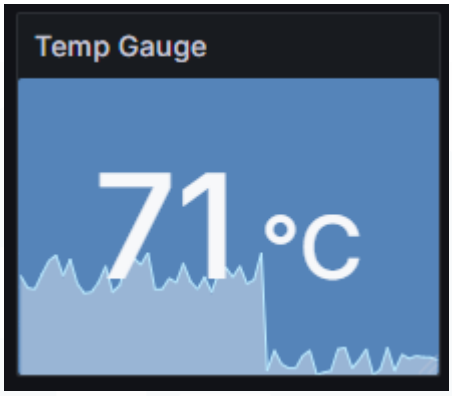
Gambar 3.9 Tampilan Grafana

Pada sisi kiri layar ditampilkan visualisasi data yang ditarik dari database. Sisi kanan layar menunjukkan daftar *peringatan*, yang menunjukkan jika kondisi mesin dalam keadaan normal atau tidak sesuai dengan parameter yang ditentukan. Perubahan pada daftar peringatan dapat diamati pada Gambar 3.10.



Gambar 3.10 Tampilan daftar peringatan jika kerja mesin abnormal

Data yang ditampilkan pada HMI dapat diverifikasi dengan mengubah nilai acak yang dibuat pada program Node-RED, dan mengamati dampaknya pada tampilan HMI. Dengan mengubah nilai node *random* untuk nilai suhu dari nilai normalnya yaitu 150 hingga 180 ke *range* lain yaitu 60-80, maka *gauge* temperatur berubah, seperti pada gambar 3.11. Maka dapat dipastikan koneksi berjalan lancar dari simulasi Node-RED hingga ke Grafana.



Gambar 3.11 Tampak data temperatur setelah perubahan pada data sumber

3.2.4 Sistem *peringatan* dengan Grafana

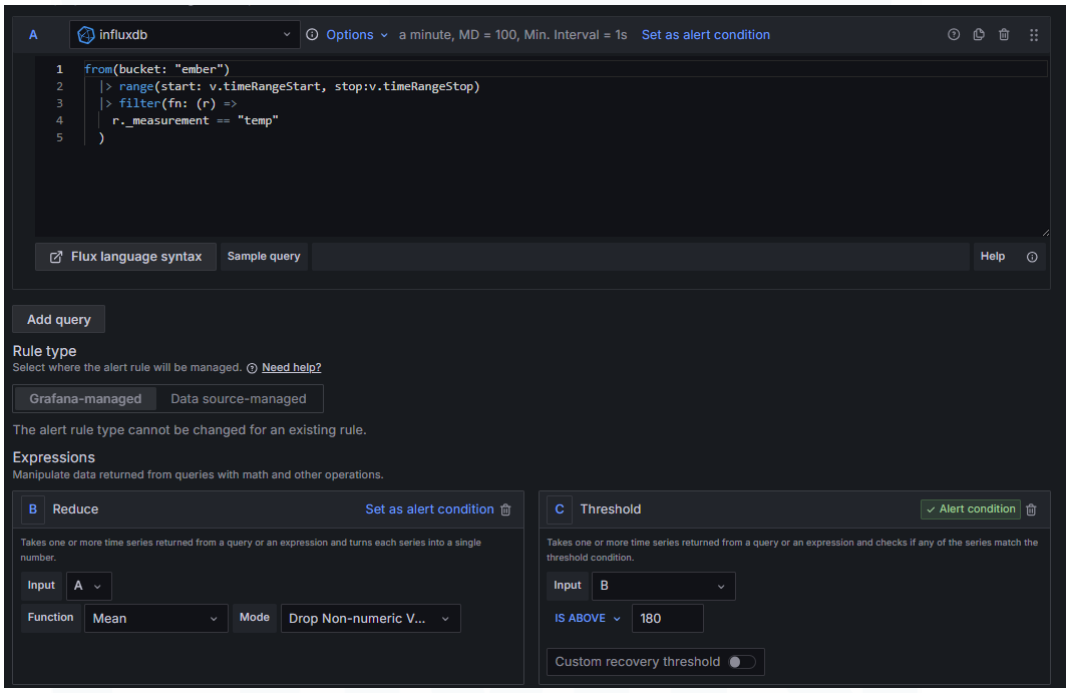
Dengan menggunakan sistem pengaturan peringatan yang tersedia pada Grafana, maka dapat ditentukan batas kerja normal pada variabel yang ditarik dari database, dan diberikan *peringatan* jika batas tersebut dilewati. Pada HMI Grafana yang dibuat, diberikan empat buah sistem peringatan yang berjalan, yaitu peringatan mengenai suhu abnormal, detektor logam, pemberhentian mesin, dan putaran motor.

Dalam pengaturan sistem peringatan, data yang didapatkan akan diproses lebih lanjut dengan beberapa jenis operasi, dan hasil dari operasi tersebut dapat dijadikan sebagai kondisi untuk menyalanya peringatan. Digunakan dua operasi utama untuk memproses data *time-series*, yaitu operasi reduksi dan *threshold*.

Pada operasi reduksi, data yang berupa data *time-series* diproses sehingga menjadi satu variabel angka saja. Dapat dilakukan beberapa jenis perhitungan untuk mendapatkan nilai tersebut, seperti mengambil nilai minimal, maksimal, rata-rata, total, atau nilai yang terakhir didapatkan dari data *time-series*.

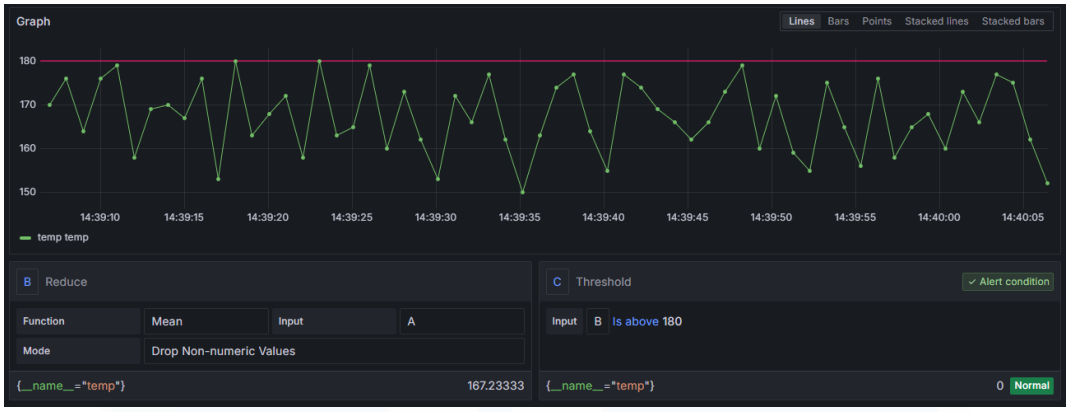
Pada operasi *threshold*, sebuah variabel akan dibandingkan ke sebuah konstanta yang ditentukan. Perbandingan ini dapat berupa lebih dari, kurang dari, dan di dalam atau luar *range* tertentu. Hasil dari operasi ini adalah nilai *Boolean* yang menunjukkan jika kondisi *threshold* tersebut terpenuhi. Nilai ini kemudian akan digunakan untuk mengaktifkan peringatan jika kondisi terpenuhi.

Pengaturan untuk kedua operasi tersebut dapat diamati pada Gambar 3.12, dengan variabel A merupakan data *time-series* dari influxDB, B merupakan nilai yang telah direduksi, dan C merupakan *boolean* yang menentukan jika alert akan menyala.



Gambar 3.12 Pengaturan operasi sistem peringatan

Dengan kedua operasi tersebut, dapat ditentukan kondisi yang tepat untuk setiap peringatan yang dibuat. Pada peringatan mengenai suhu abnormal, nilai rata-rata pengukuran suhu selama 1 menit terakhir akan dibandingkan dengan nilai konstan yaitu 180°C, dan peringatan akan aktif jika nilai yang telah diproses melewati nilai yang telah ditentukan tersebut. Operasi ini dapat dilihat pada Gambar 3.13.

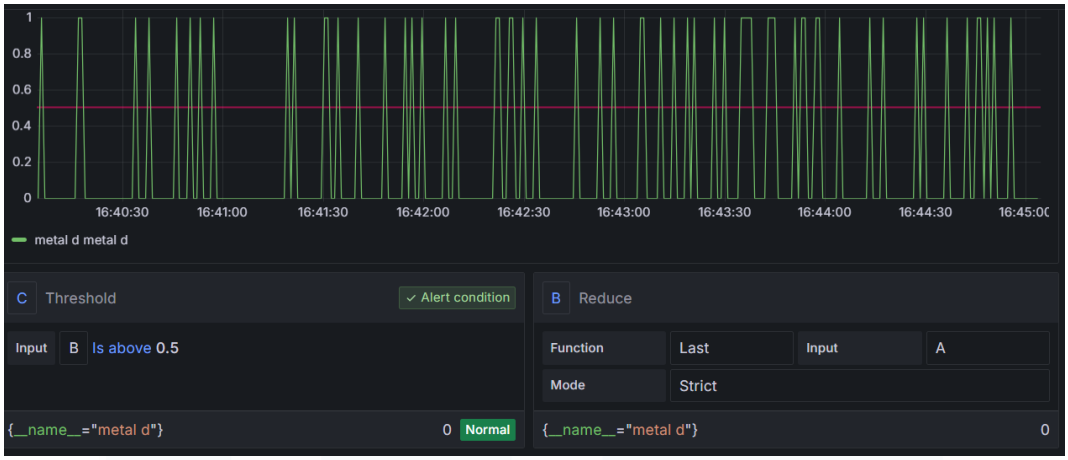


Gambar 3.13 Operasi untuk menentukan aktifnya peringatan suhu tinggi

Pada peringatan mengenai putaran motor, nilai rata-rata dari pengukuran kecepatan selama 1 menit terakhir akan dibandingkan dengan *range* operasi normal seperti pada Tabel 3.1, yaitu 2400-3600 rpm. Peringatan akan aktif jika nilai rata-rata berada di luar *range* tersebut. Operasi ini dapat dilihat pada Gambar 3.14.

Gambar 3.14 Operasi untuk menentukan aktifnya peringatan putaran motor

Pada peringatan mengenai pendeteksi logam, ditentukan bahwa jika tidak terdeteksi logam maka nilai yang akan diberikan oleh sensor adalah 0, dan 1 jika terdeteksi logam. peringatan akan aktif jika nilai terakhir yang terbaca adalah 1. Operasi ini dapat dilihat pada Gambar 3.15.



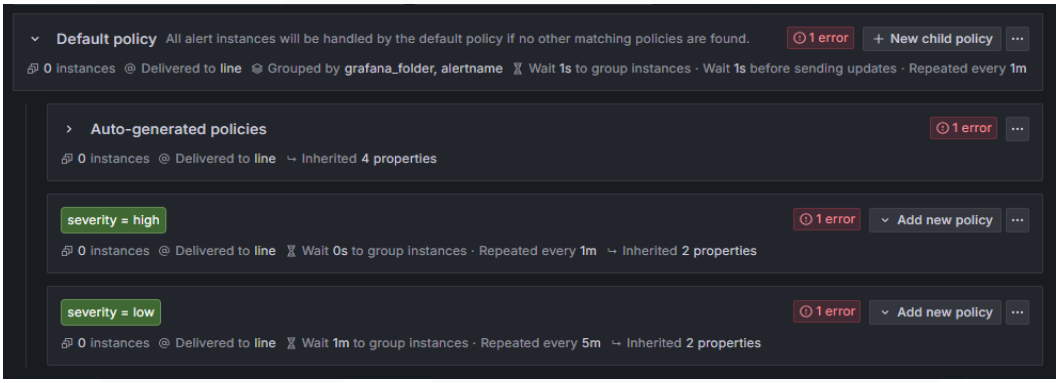
Gambar 3.15 Operasi untuk menentukan aktifnya peringatan pendeteksi logam

Peringatan mengenai pemberhentian mesin bekerja serupa dengan peringatan untuk mendeteksi logam, dimana peringatan akan aktif jika sensor arus listrik mendeteksi bahwa mesin sedang dalam keadaan mati (listrik tidak mengalir). Kondisi keaktifan mesin akan dikompilasi selama 1 jam terakhir, dan peringatan akan aktif jika mesin menyala kurang dari 30% dari waktu yang berjalan. Operasi ini dapat dilihat pada Gambar 3.16.

Gambar 3.16 Operasi untuk menentukan aktifnya peringatan pemberhentian mesin

Peringatan yang sedang dalam kondisi aktif akan diproses lebih lanjut dengan pengaturan parameter notifikasi, yang berfungsi untuk mengirim pesan melalui media tertentu tentang peringatan tersebut. Parameter notifikasi dapat diatur untuk mengirimkan pesan mengenai peringatan yang aktif atau terselesaikan. Notifikasi dapat dikirim ulang jika peringatan tetap aktif setelah sebuah jangka waktu setelah

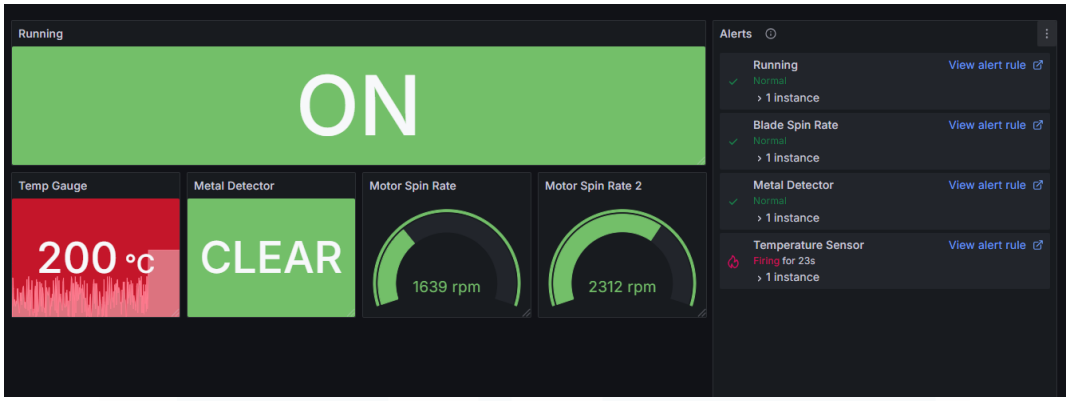
notifikasi pertama terkirim. Jangka waktu ini dapat diatur, seperti pada Gambar 3.17.



Gambar 3.17 Pengaturan notifikasi

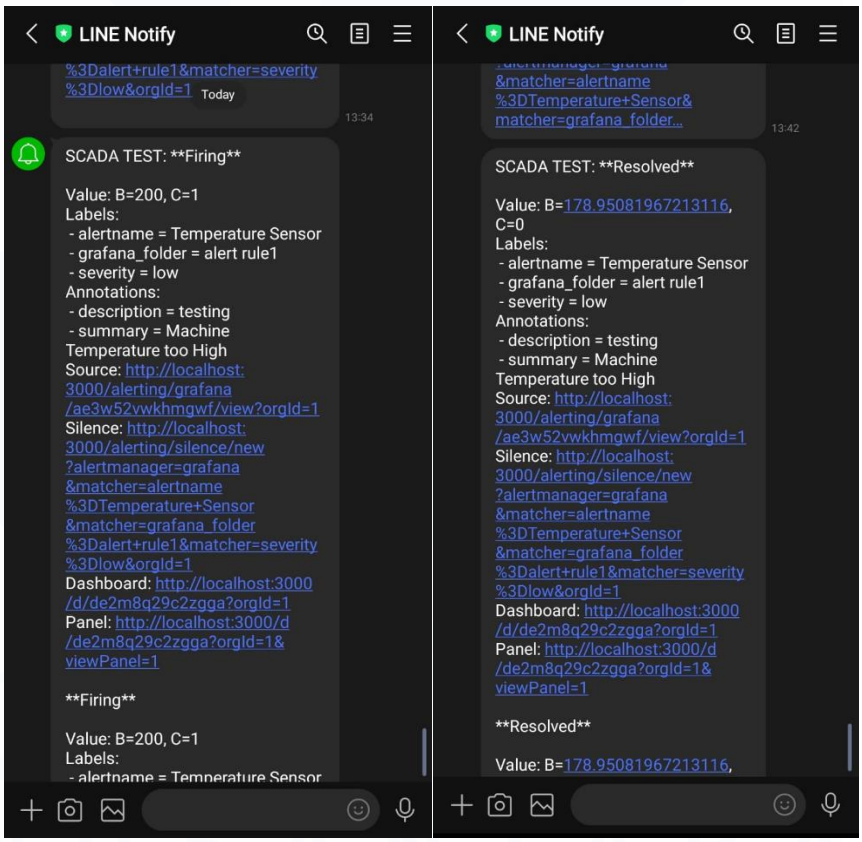
Dalam sistem notifikasi yang dibuat, peringatan dibagi menjadi dua kategori, yaitu kategori cepat (*high severity*), dan kategori lambat (*low severity*). Pada kategori cepat, notifikasi akan langsung dikirim setelah peringatan aktif, dan pengiriman ulang dilakukan setelah 1 menit peringatan tetap aktif. Pada kategori lambat, notifikasi akan dikirim 1 menit setelah peringatan aktif, dan mengulang setiap 5 menit setelahnya. Peringatan pendeteksi logam, kecepatan motor, dan temperatur dikelompokkan dalam kategori cepat, dan peringatan pemberhentian mesin dimasukkan ke kategori lambat.

Untuk verifikasi sistem notifikasi yang dibuat, pengujian dilakukan dengan mengatur nilai pada program Node-RED untuk melebihi nilai *threshold* yang telah diatur. Sebagai contoh, nilai temperatur diatur menjadi temperatur konstan 200°C untuk menyimulasikan kondisi *overheat.* Dalam kondisi ini, peringatan temperature pada HMI menyala seperti pada gambar 3.18, dan pesan peringatan dapat dikirimkan dengan sukses.



Gambar 3.18 Tampilan HMI ketika kondisi temperatur terlalu tinggi

Setelah pengaturan temperatur dikembalikan ke pengaturan semula dan rata-rata temperatur turun ke bawah *threshold* yang telah ditentukan, peringatan kembali ke kondisi tidak aktif, dan pesan kedua dapat dikirimkan untuk memberitahu bila kondisi sudah kembali normal. Notifikasi yang dikirim dapat dilihat pada Gambar 3.19.



Gambar 3.19 Tampilan notifikasi pada aplikasi LINE

3.3 Kendala yang Ditemukan

Dalam proses pembuatan HMI dan sistem notifikasi, ditemukan beberapa kendala yang mempersulit proses pengerjaan.

Mesin *wood chipper* yang digunakan berada di lokasi yang jauh dari kantor tempat pengerjaan proyek. Maka, selama pengerjaan berlangsung, mesin tersebut tidak dapat diamati secara langsung. HMI juga tidak dapat diuji dengan menggunakan *hardware* sesungguhnya.

Dalam proses pembuatan sistem notifikasi, terdapat beberapa jenis media yang dapat digunakan untuk mengirimkan notifikasi. Salah satu kandidat media yang dapat digunakan untuk mengirim peringatan adalah melalui email. Namun, pengiriman notifikasi melalui email memerlukan pengaturan pada server internal perusahaan yang mana server tersebut tidak dapat diakses oleh peserta magang.

3.4 Solusi atas Kendala yang Ditemukan

Untuk dapat menjawab kedua kendala yang ditemukan, diberlakukan solusi sebagai berikut:

Untuk mengerjakan HMI, keseluruhan mesin *wood chipper* dapat direduksi menjadi beberapa variabel penting yang ingin diamati. Variabel-variabel tersebut kemudian disimulasikan dengan menggunakan Node-RED.

Notifikasi pada sistem yang dibuat menggunakan media alternatif, yaitu dengan melalui media sosial LINE. Dengan menggunakan fitur LINE Notify, dapat dibuat bot yang mengirimkan pesan notifikasi.