

## **BAB III**

### **Pelaksanaan Kerja Magang**

#### **3.1. Kedudukan dan Koordinasi**

Selama melaksanakan program magang di PT SATSINDO, posisi *Junior Engineer* dilaksanakan di bawah supervisi langsung Michael Bryan Jahanto sebagai *Senior Software Engineer*. Dalam pelaksanaan tugasnya, pengembangan sistem berbasis AI dan juga terlibat dalam berbagai tugas pendukung yang bersifat multidisiplin untuk mendukung kebutuhan operasional dan pengembangan kompetensi teknis secara menyeluruh.

Tugas utama yang dilakukan adalah merancang dan mengimplementasikan Sistem Darurat Keamanan Berbasis AI yang mampu mendeteksi pelanggaran area aman di lingkungan industri dan mengambil tindakan otomatis berupa peringatan (*warning*) atau penghentian operasi mesin (*shutdown*). Selain tugas utama tersebut, tanggung jawab tambahan juga diberikan dalam kegiatan edukasi dan pelatihan dasar PLC bagi staff operasional di pabrik-pabrik, serta terlibat dalam perancangan dan perakitan panel listrik 3 *phase* yang digunakan dalam lingkungan produksi dan mempelajari dan mempraktikkan sistem pengabelan dan tata letak kabel dalam panel kontrol.

Koordinasi dan bimbingan teknis dilakukan secara rutin dengan supervisor dan tim engineer lainnya untuk memastikan keselarasan antara desain sistem dan implementasi AI. Proses diskusi dan *review* dilakukan secara berkala untuk mengevaluasi kemajuan proyek, mengatasi kendala teknis, serta memastikan bahwa seluruh komponen sistem dapat terintegrasi dengan baik sesuai dengan spesifikasi dan standar keamanan yang berlaku.

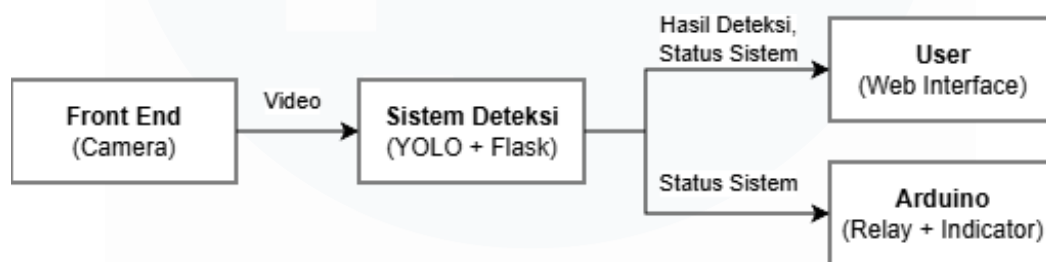
#### **3.2. Tugas dan Uraian Kerja Magang**

Proyek utama berfokus pada pengembangan Sistem Darurat Keamanan Berbasis AI untuk Penghentian Operasi Mesin Industri secara Otomatis. Proyek ini dirancang untuk meningkatkan keselamatan kerja di area industri, khususnya pada

jalur produksi yang memiliki potensi risiko kecelakaan akibat interaksi manusia dan mesin.

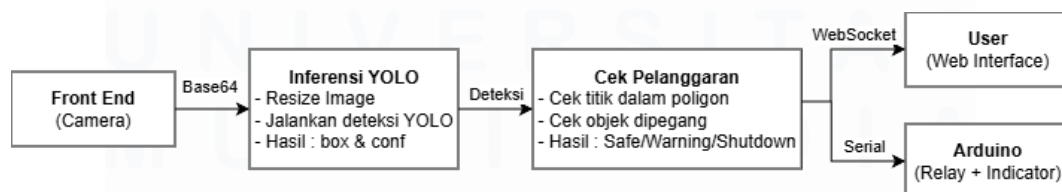
### 3.2.1. Pengembangan Sistem Backend

*Backend* pada sistem ini dibangun menggunakan *FastAPI* sebagai framework web asinkron yang mendukung komunikasi HTTP dan *WebSocket*. *Backend* bertanggung jawab untuk menjalankan dua model YOLO secara simultan: YOLO Pose untuk deteksi pose manusia dan YOLO Object Detection untuk mendeteksi objek tertentu yang relevan dengan lingkungan industri. Selain itu, backend juga mengelola komunikasi dengan Arduino melalui *endpoint REST API* dan menyediakan *WebSocket* untuk streaming data deteksi secara *real-time* kepada frontend.



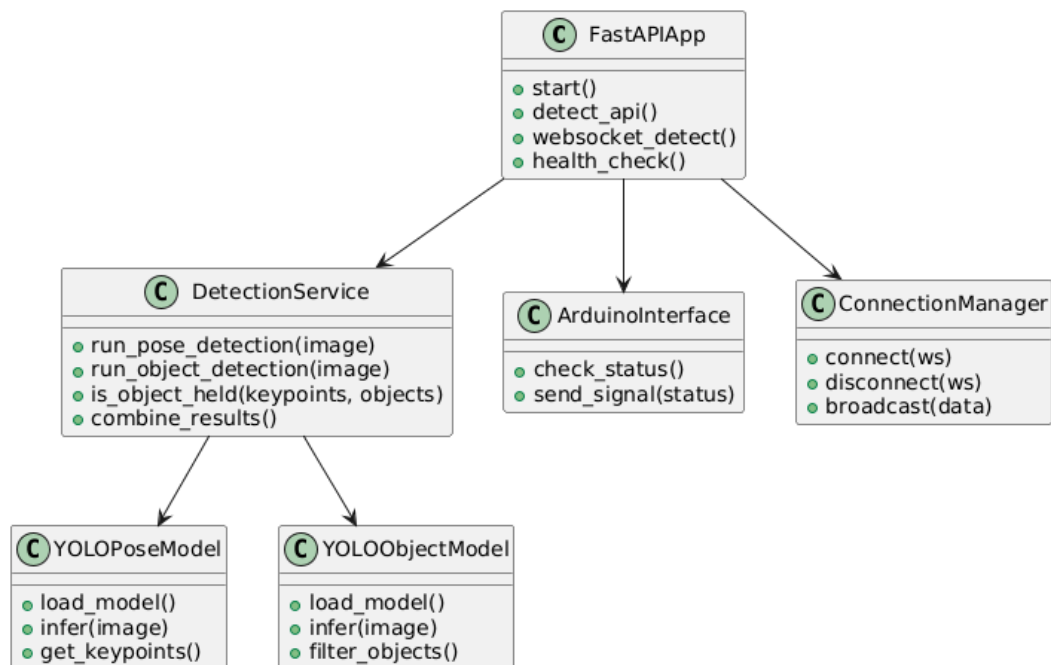
Gambar 3.1 DFD Level 0

Secara konseptual, arsitektur sistem pada Gambar 3.1 merepresentasikan sistem deteksi keselamatan sebagai satu proses utama yang menerima aliran data berupa video/frame dari kamera (frontend), kemudian menghasilkan keluaran berupa status sistem dan hasil deteksi yang dikirimkan ke antarmuka pengguna (web interface) serta sinyal kontrol ke Arduino (relay dan indikator). Backend berperan sebagai pusat pemrosesan yang mengintegrasikan input visual, logika deteksi, dan aktuasi perangkat keras.



Gambar 3.2 DFD Level 1

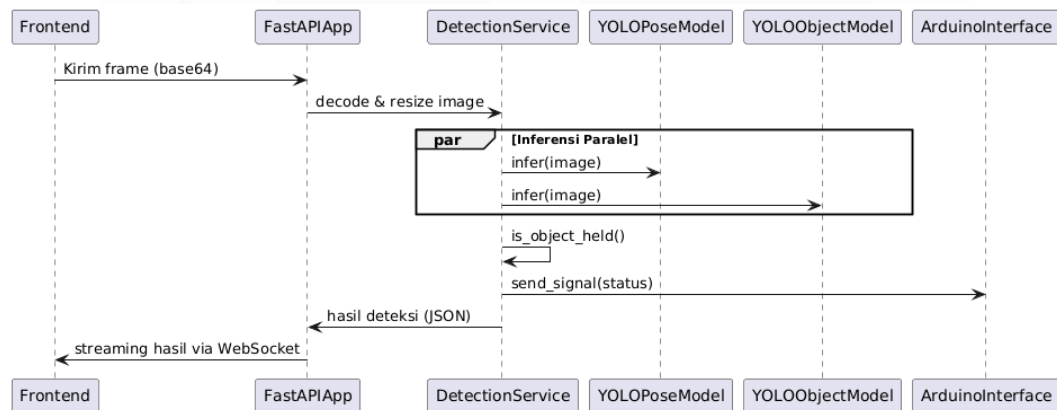
Arsitektur *backend* diorganisasikan dalam beberapa modul utama: FastAPI Application yang mengelola HTTP dan *WebSocket*, YOLO Pose Model yang menggunakan YOLOv11-pose untuk mendeteksi 17 *keypoints* manusia, YOLO Object Detection Model dengan bobot COCO untuk mendeteksi 80 kelas objek dengan filter untuk objek tertentu seperti bottle, knife, scissors, laptop, Arduino Interface untuk mengirim sinyal ke Arduino berdasarkan status sistem (SAFE, WARNING, SHUTDOWN), dan WebSocket Manager yang menangani koneksi WebSocket untuk streaming hasil deteksi secara real-time.



Gambar 3.3 UML Class Diagram

UML Class Diagram pada gambar 3.3 menunjukkan bahwa *FastAPIApp* bertindak sebagai *entry point* utama dengan fungsi *start()*, *detect\_api()*, *websocket\_detect()*, dan *health\_check()*. *FastAPIApp* berinteraksi dengan *DetectionService* yang bertanggung jawab menjalankan logika inti deteksi, termasuk menjalankan inferensi pose dan objek, menentukan apakah objek sedang dipegang (*is\_object\_held()*), serta menggabungkan hasil deteksi (*combine\_results()*). *DetectionService* memanfaatkan dua kelas model terpisah, yaitu *YOLOPoseModel* untuk mendeteksi 17 *keypoints* manusia dan *YOLOObjectModel* untuk mendeteksi objek menggunakan bobot COCO dengan

penyaringan objek relevan. Selain itu, *ArduinoInterface* menangani pengiriman sinyal status sistem ke perangkat Arduino, sedangkan *ConnectionManager* mengelola seluruh koneksi *WebSocket* aktif untuk kebutuhan *streaming* data *real-time*.



Gambar 3.4 UML Sequence Diagram

Alur eksekusi sistem diperjelas melalui *UML Sequence Diagram*, yang menggambarkan cara kerja sistem secara keseluruhan berdasarkan urutan waktu dan kejadian yang dilakukan oleh *software*.

### 3.2.2. Pengembangan Sistem Frontend

Frontend dibangun sebagai *single-page application* (SPA) menggunakan HTML5, CSS3, dan JavaScript. Arsitektur *frontend* terdiri dari *Video Streaming Module* yang menangani *capture video* dari kamera menggunakan *WebRTC* (*getUserMedia*), *WebSocket Client* yang terhubung ke *backend* melalui *WebSocket* untuk menerima data deteksi, *Canvas Drawing Module* dengan 3 lapis canvas untuk *video stream*, *overlay* zona keselamatan, dan *overlay skeleton* serta *bounding box*, *Zone Management Module* untuk menggambar, menyimpan, dan memuat zona keselamatan (*polygon*), *UI Control Panel* untuk admin (login, pengaturan zona, TTS, dll.), dan *Event Logging System* yang mencatat setiap kejadian pelanggaran dengan *screenshot*.

Mekanisme *WebSocket* pada frontend diimplementasikan dengan fungsi *connectWebSocket()* yang membuka koneksi ke *backend WebSocket*

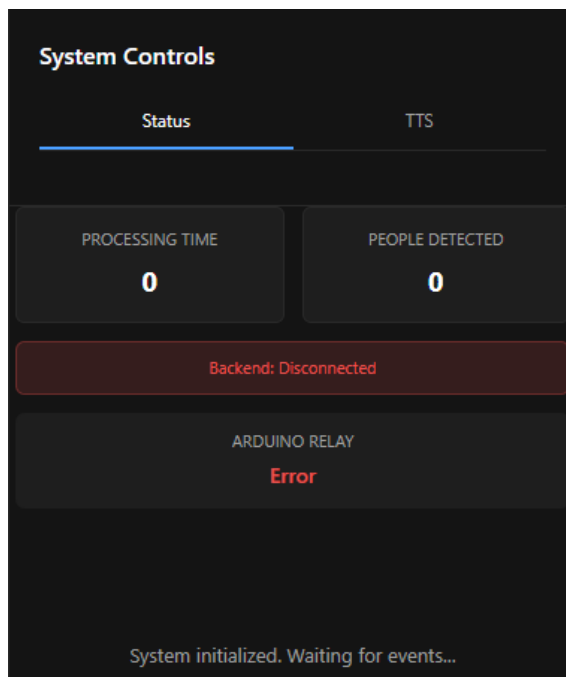
*endpoint*. *Frontend* secara periodik (setiap 100ms) menangkap frame video, mengonversinya ke *base64*, dan mengirimkannya melalui *WebSocket*. Koneksi *WebSocket* dikelola dengan mekanisme *reconnection* otomatis jika terputus. Data yang diterima dari *WebSocket* berisi hasil deteksi pose dan objek yang kemudian diproses untuk visualisasi pada *canvas*. *WebSocket* juga digunakan untuk menerima status sistem (*safe*, *warning*, *shutdown*) secara *real-time* yang kemudian ditampilkan pada antarmuka pengguna.



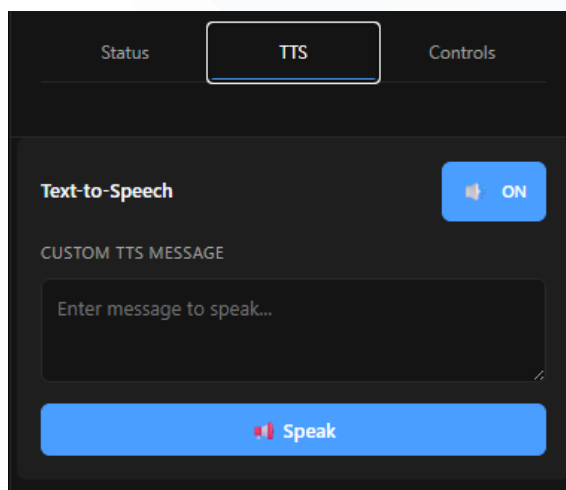
Gambar 3.5 Halaman *Factory Safety AI Monitor*

Tampilan utama sistem *frontend* ditunjukkan pada Gambar 3.4. Secara umum, *frontend* menampilkan *live streaming* kamera sebagai sumber data utama. *Frontend* secara periodik (setiap 100ms) menangkap frame video, mengonversinya ke *base64*, dan mengirimkannya melalui *WebSocket*. Koneksi *WebSocket* dikelola dengan mekanisme *reconnection* otomatis jika terputus.

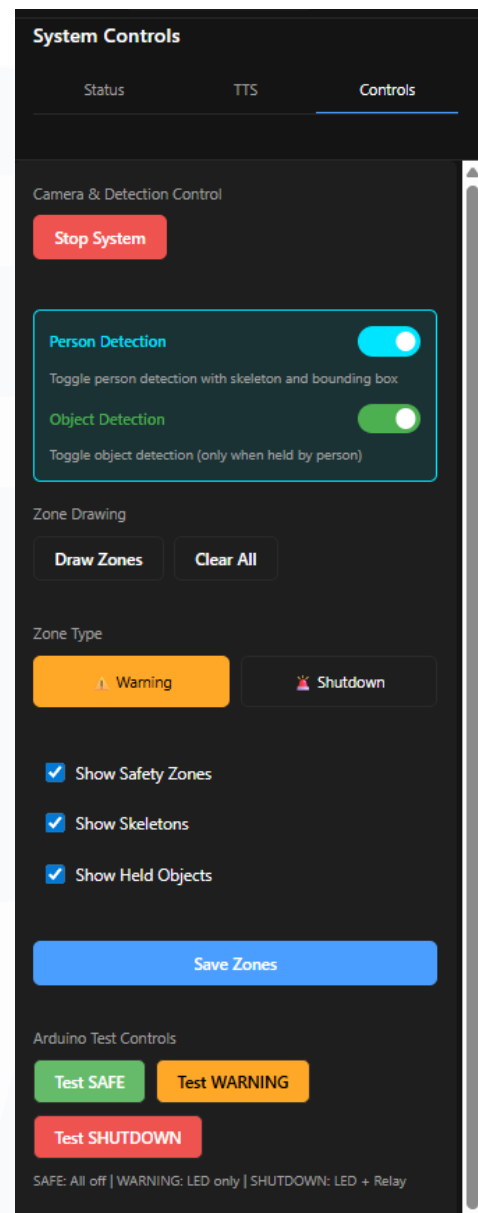
Dalam sistem ini terdapat dua jenis peran pengguna, yaitu user dan admin. User didefinisikan sebagai pengguna yang hanya memiliki hak akses untuk memantau sistem, melihat hasil deteksi, status keselamatan, dan peringatan yang ditampilkan. Admin merupakan pengguna dengan hak akses penuh untuk melakukan konfigurasi sistem, termasuk penggambaran zona keselamatan, penghapusan zona, pengelolaan pesan TTS, serta pengelolaan log kejadian.



[A]



[B]



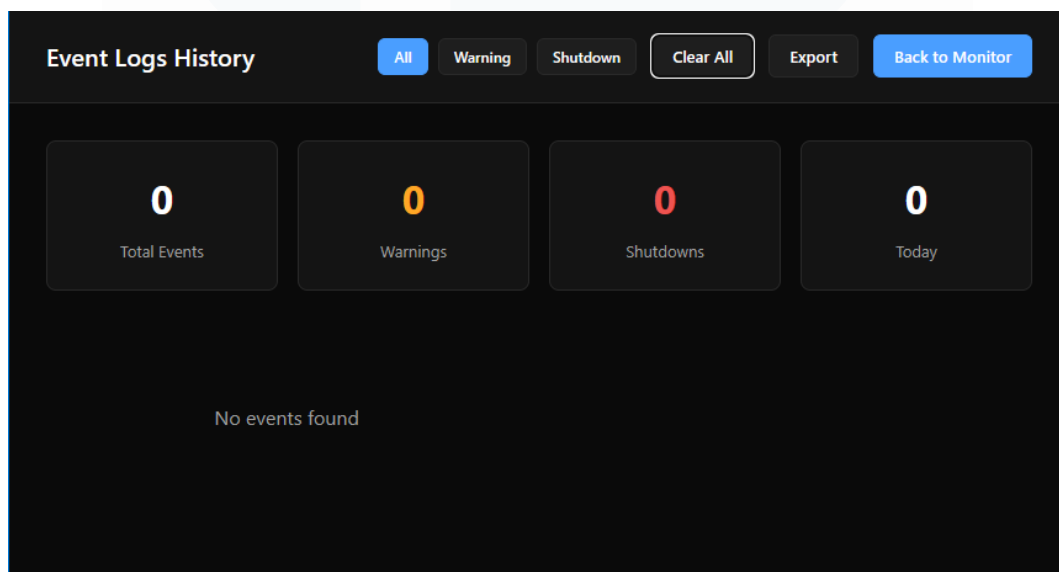
[C]

Gambar 3.6 [A] Modul Status, [B] Modul TTS, [C] Modul Kontrol

Frontend terdiri dari beberapa modul utama. Modul live video dan visualisasi deteksi menampilkan *bounding box* dan zona keselamatan di atas video, yang merupakan representasi langsung dari hasil inferensi dan evaluasi zona yang dilakukan oleh *backend*. Modul status monitoring yang ditunjukkan pada Gambar 3.5 bagian [A] menampilkan informasi kondisi backend, status Arduino, kecepatan pemrosesan (FPS), dan waktu inferensi.

Modul Text to Speech (TTS) yang ditampilkan pada Gambar 3.5 bagian [B] berfungsi memberikan peringatan suara otomatis ketika backend mengirimkan status *warning* atau *shutdown*. TTS digunakan sebagai sistem peringatan tambahan untuk meningkatkan kewaspadaan operator, terutama pada lingkungan industri yang memiliki keterbatasan visual atau tingkat kebisingan tinggi. Sistem juga menyediakan kolom *Custom Message* yang memungkinkan admin menentukan pesan suara khusus sesuai kebutuhan operasional.

Modul Controls yang ditunjukkan pada Gambar 3.5 bagian [C] hanya dapat diakses oleh pengguna dengan hak akses admin. Modul ini mencakup fitur *Zone Drawing* dan *Clear All Zones*. Fitur *Zone Drawing* memungkinkan admin menggambar zona keselamatan berbentuk poligon langsung pada tampilan video. Zona yang digambar pada frontend ini kemudian dikirimkan ke backend dan digunakan sebagai parameter evaluasi pelanggaran pada setiap proses deteksi.



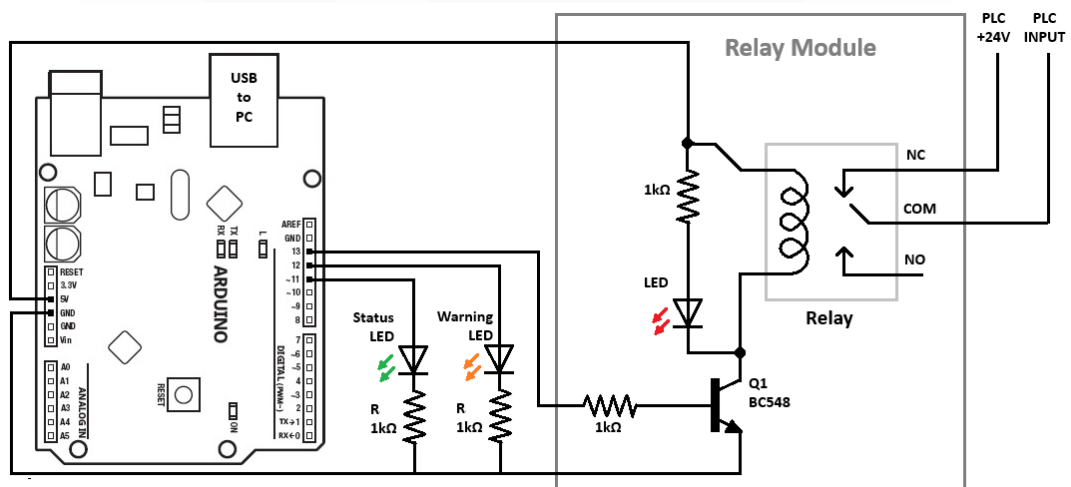
Gambar 3.7 Fitur *Event Logging*

Frontend juga menyediakan fitur event logging yang ditampilkan pada Gambar 3.6. Fitur ini mencatat setiap kejadian pelanggaran zona secara otomatis, lengkap dengan waktu kejadian, jenis pelanggaran, dan tangkapan layar sebagai bukti visual. Admin dapat menghapus seluruh riwayat kejadian atau mengekspor data dalam format JSON untuk keperluan dokumentasi dan evaluasi.

### 3.2.3. Pengembangan Sistem Fisik

Pengembangan sistem fisik pada *Factory Safety AI Monitor* bertujuan merealisasikan keluaran keputusan sistem backend ke dalam bentuk aksi nyata pada lingkungan industri. Sistem fisik berfungsi sebagai penghubung antara keputusan logika berbasis kecerdasan buatan dan sistem kontrol mesin industri yang umumnya menggunakan PLC. Pada tahap ini, Arduino digunakan sebagai *interface controller* yang menerima perintah dari backend dan menerjemahkannya menjadi sinyal listrik untuk mengendalikan modul relay serta indikator visual berupa LED.

Secara umum, sistem fisik terdiri dari tiga komponen utama, yaitu Arduino sebagai pengendali logika, modul relay sebagai penghubung ke sistem PLC, dan rangkaian indikator LED sebagai penanda status sistem. Konfigurasi keseluruhan sistem fisik dapat dilihat pada Gambar 3.7, yang memperlihatkan hubungan antara Arduino, rangkaian driver relay, dan input PLC.



Gambar 3.8 Wiring Diagram Sistem fisik

Arduino berfungsi sebagai *slave controller* yang menerima perintah dari backend melalui komunikasi serial USB berdasarkan evaluasi zona keselamatan. Perintah seperti *SAFE*, *WARNING*, atau *SHUTDOWN* diproses Arduino untuk mengatur kondisi pin output digital sehingga keputusan *backend* berdampak langsung pada sistem fisik.

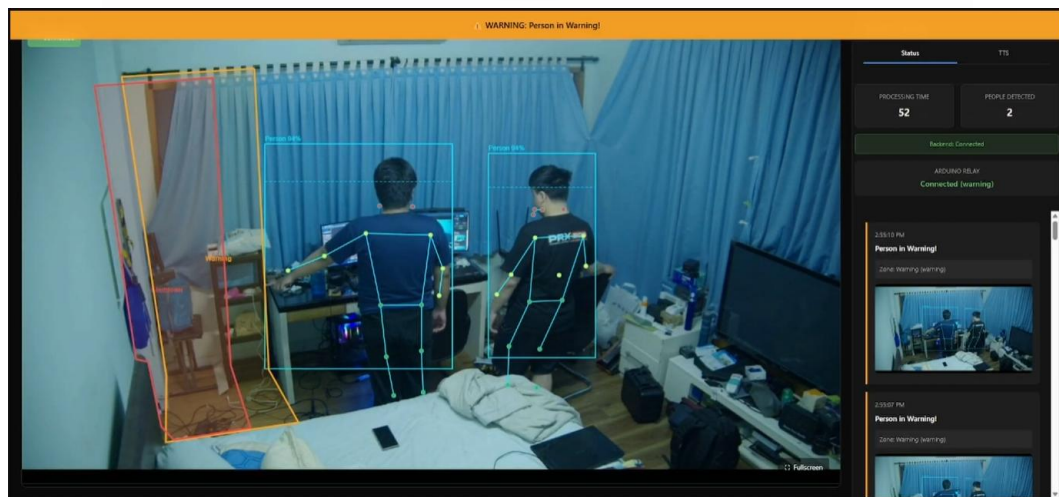


Arduino mengendalikan dua indikator LED, yaitu Status LED untuk kondisi aman dan Warning LED untuk kondisi peringatan atau darurat. Selain LED, Arduino juga mengontrol modul relay. Relay terhubung ke input PLC menggunakan kontak NC, dan COM. Pada kondisi aman, relay berada di posisi NC sehingga PLC tetap aktif. Ketika perintah *shutdown* diterima, relay aktif dan memutus atau mengalihkan sinyal PLC sesuai konfigurasi keselamatan.

### 3.2.4. Pengujian Sistem

Tahap pengujian sistem dilakukan untuk memastikan bahwa seluruh komponen *Factory Safety AI Monitor*, khususnya model deteksi berbasis YOLO, mampu bekerja secara akurat, konsisten, dan responsif dalam mendeteksi keberadaan manusia serta memicu aksi keselamatan pada sistem fisik.

Pengujian deteksi orang pada zona Warning dilakukan dengan mendefinisikan area zona peringatan pada lingkungan pengujian menggunakan fitur zone drawing pada antarmuka web. Zona warning ditandai dengan poligon berwarna kuning/oranye.

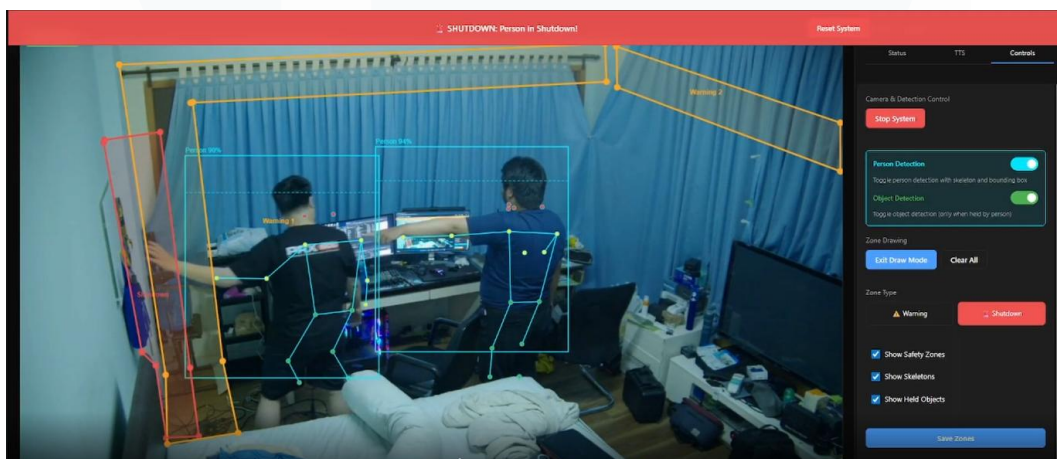


Gambar 3.9 Pengujian Zona Warning

Berdasarkan hasil pengujian, ketika seseorang memasuki area zona warning, sistem berhasil mendeteksi keberadaan orang tersebut menggunakan YOLO Pose Model. Deteksi ditunjukkan dengan munculnya bounding box bertanda “Person” beserta skeleton pose yang terdiri dari 17 *keypoints*. Pada saat yang sama, sistem

menampilkan notifikasi “*WARNING: Person in Warning!*” pada bagian atas antarmuka. Selain itu, status sistem pada panel informasi berubah menjadi Warning, dan sinyal peringatan dikirimkan ke Arduino untuk mengaktifkan indikator sesuai dengan kondisi warning. Hasil ini menunjukkan bahwa sistem mampu mendeteksi posisi orang secara akurat terhadap batas zona warning yang telah ditentukan.

Pengujian selanjutnya dilakukan pada zona *Shutdown*, yang merupakan zona dengan tingkat risiko tertinggi. Zona *shutdown* digambarkan menggunakan poligon berwarna merah. Tujuan pengujian ini adalah untuk memastikan bahwa sistem dapat mendeteksi pelanggaran zona berbahaya dan memberikan respons yang lebih tegas dibandingkan zona warning.



Gambar 3.10 Pengujian Zona Shutdown

Hasil pengujian menunjukkan bahwa ketika seseorang memasuki zona shutdown, sistem secara langsung mengubah status menjadi *SHUTDOWN*. Notifikasi “*SHUTDOWN: Person in Shutdown!*” ditampilkan pada antarmuka pengguna, dan sistem secara otomatis mengirimkan sinyal shutdown ke Arduino untuk menyalakan relay untuk mematikan PLC.

Pengujian deteksi objek yang dipegang dilakukan untuk mengevaluasi kemampuan sistem dalam mengidentifikasi interaksi antara manusia dan objek. Pada pengujian ini, objek berupa laptop digunakan sebagai contoh objek yang relevan.



Gambar 3.11 Pengujian Benda

Hasil pengujian menunjukkan bahwa YOLO *Object Detection Model* berhasil mendeteksi objek laptop dengan *confidence score* yang ditampilkan pada antarmuka (misalnya “*laptop 84%*”). Selanjutnya, sistem memeriksa hubungan antara objek dan manusia menggunakan fungsi *is\_object\_held()*, yang menghitung jarak *Euclidean* antara *keypoints* pergelangan tangan manusia dengan pusat bounding box objek serta memeriksa apakah objek berada di dalam *bounding box* manusia. Ketika objek berada dalam jarak ambang (*HOLD\_DISTANCE\_THRESHOLD*) atau berada di dalam *bounding box* manusia, sistem menandai objek tersebut sebagai *HELD*. Informasi ini ditampilkan secara visual pada antarmuka dan dapat digunakan sebagai parameter tambahan dalam penentuan status keselamatan. Hasil ini membuktikan bahwa sistem tidak hanya

mampu mendeteksi keberadaan objek, tetapi juga memahami konteks interaksi antara manusia dan objek tersebut.



Gambar 3.12 Pegujian Sistem Fisik

Hasil pengujian tersebut menunjukkan bahwa model deteksi dan sistem pendukung mampu bekerja secara cepat dan andal untuk kebutuhan keselamatan industri. Dengan waktu respons di bawah setengah detik, sistem dinilai cukup responsif untuk mendeteksi potensi bahaya dan melakukan tindakan pencegahan secara otomatis. Hal ini membuktikan bahwa model hasil pelatihan serta integrasinya dengan backend dan sistem fisik telah memenuhi kebutuhan operasional sistem keselamatan berbasis kecerdasan buatan.

### 3.3. Kendala yang Ditemukan

Selama pelaksanaan pengembangan *Factory Safety AI Monitor*, terdapat beberapa kendala yang memengaruhi proses pengembangan dan pengujian sistem, khususnya pada tahap pelatihan dan implementasi model deteksi berbasis YOLO. Salah satu kendala utama yang ditemukan adalah keterbatasan perangkat komputasi yang digunakan pada tahap awal pengembangan. Perangkat percobaan yang tersedia memiliki kemampuan pemrosesan yang terbatas, sehingga proses pelatihan model dan pengujian inferensi membutuhkan waktu yang relatif lama. Kondisi ini menyebabkan iterasi pelatihan model tidak dapat dilakukan secara cepat, terutama ketika diperlukan penyesuaian parameter atau penambahan data latih.

Kendala berikutnya berkaitan dengan kondisi lingkungan industri yang menjadi objek pengamatan sistem. Setiap area pabrik memiliki karakteristik yang berbeda, baik dari segi tata letak, sudut pengambilan gambar kamera CCTV, maupun kondisi pencahayaan. Pencahayaan yang tidak selalu optimal, seperti area dengan intensitas cahaya rendah, bayangan tajam, atau pantulan cahaya berlebih, menyebabkan variasi kualitas citra yang cukup signifikan. Hal ini berdampak pada kesulitan dalam mengumpulkan dan mengolah data latih yang representatif.

Selain itu, data pelatihan yang bersumber dari rekaman CCTV memiliki tingkat *noise* yang relatif tinggi, seperti resolusi yang rendah, sudut pandang yang tidak ideal, serta adanya objek lain yang menyerupai bentuk manusia. Kondisi ini menyebabkan proses pelabelan data dan pelatihan model menjadi lebih kompleks dan memerlukan beberapa kali proses pelatihan ulang untuk memperoleh hasil deteksi yang optimal dan stabil.

### **3.4. Solusi atas Kendala yang Ditemukan**

Untuk mengatasi keterbatasan perangkat komputasi pada tahap pelatihan model, solusi yang diterapkan adalah penggunaan perangkat dengan spesifikasi yang lebih tinggi, yaitu kartu grafis NVIDIA GeForce RTX 4070 secara remote. Penggunaan GPU dengan kemampuan komputasi yang lebih besar secara signifikan mempercepat proses pelatihan dan pengujian model, sehingga iterasi pelatihan dapat dilakukan lebih sering dan efisien. Dengan dukungan perangkat ini, proses *training* yang sebelumnya memakan waktu lama dapat dipersingkat, sehingga pengembangan model menjadi lebih optimal.

Dalam menghadapi variasi kondisi lingkungan pabrik dan pencahayaan yang tidak konsisten, dilakukan penyesuaian pada proses pengumpulan dan pelatihan data. Dataset diperluas dengan menambahkan variasi kondisi pencahayaan, sudut kamera, dan jarak objek untuk meningkatkan kemampuan generalisasi model. Selain itu, proses pelatihan dilakukan secara bertahap dengan beberapa kali *re-training* hingga diperoleh model yang mampu mendeteksi objek manusia secara lebih stabil pada berbagai kondisi lingkungan.

Untuk mengatasi kesulitan dalam pengolahan data CCTV, dilakukan proses kurasi dan seleksi data yang lebih ketat, termasuk penghapusan data yang terlalu buram atau tidak relevan, serta penyesuaian anotasi agar lebih konsisten. Melalui pendekatan ini, kualitas data latih dapat ditingkatkan sehingga model hasil pelatihan mampu memberikan performa deteksi yang lebih baik dan sesuai dengan kebutuhan sistem keselamatan industri.

