BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Pelaksanaan kegiatan magang di KinetixPro Pte. Ltd. berada di dalam Divisi *Quality Assurance*, dengan penempatan pada posisi *Quality Assurance Engineer Intern*. Berdasarkan struktur organisasi perusahaan, Divisi *Quality Assurance* merupakan salah satu unit yang berada dalam lingkup koordinasi *Chief Technology Officer* (CTO), dan bekerja secara lintas fungsi bersama tim *Product Developer* serta *Artificial Intelligence & Machine Learning* (AI/ML).

Selama periode magang, Divisi *Quality Assurance* diisi oleh satu personel dan tergabung dalam tim inti lintas divisi yang terdiri dari 7 anggota tetap. Tim inti ini mencakup personel dari Divisi *Product Development*, AI/ML, dan *Quality Assurance* yang bekerja sama dalam pengembangan serta pengujian produk dan sistem kecerdasan buatan perusahaan. Pelaksanaan kegiatan magang bersifat kolaboratif dan difokuskan pada dukungan terhadap proses pengujian kualitas serta evaluasi performa sistem yang dikembangkan.

Arah dan koordinasi proyek dipimpin langsung oleh *Chief Executive Officer* (CEO), Bapak Joel Lee, serta *Chief Technology Officer* (CTO), Bapak Kun Hao Yeh. Pada awal masa magang, bimbingan teknis diberikan oleh Bapak Thomas Clark selaku *AI/ML Engineer*. Selanjutnya, kegiatan dijalankan secara mandiri dengan kontribusi aktif dalam setiap proses kerja tim lintas divisi.

Seluruh kegiatan kerja dilaksanakan secara jarak jauh (*full remote*) karena KinetixPro Pte. Ltd. berbasis di Singapura dan menerapkan sistem kerja daring penuh. Proses koordinasi dilakukan melalui *platform* digital seperti Discord, Jira, Google Meet, dan Microsoft Teams. Rapat formal dijadwalkan 2 kali setiap minggu untuk membahas perkembangan proyek, sementara komunikasi tambahan dilakukan secara fleksibel melalui pesan langsung atau panggilan pribadi (*one-on-one*) sesuai kebutuhan. Pola koordinasi ini mendukung fleksibilitas kerja sekaligus menjaga efektivitas komunikasi dan kolaborasi dalam lingkungan kerja virtual yang dinamis.

3.2 Tugas yang Dilakukan

Selama masa magang sebagai *Quality Assurance Engineer Intern*, berbagai aktivitas penting telah dilaksanakan guna mendukung pengembangan dan pengujian sistem kecerdasan buatan (AI) di perusahaan. Seluruh proses dilakukan secara terstruktur dan terkoordinasi bersama tim lintas fungsi untuk memastikan kualitas data dan performa model AI sesuai dengan standar yang ditetapkan. Adapun tahapan utama yang dilakukan adalah sebagai berikut:

1. Mengumpulkan data berupa video dan gambar sebagai sumber utama dalam pembangunan *dataset*.
2. Melakukan proses pelabelan secara manual maupun otomatis menggunakan metode *autolabel*. Untuk pelabelan manual, menyerahkan sebagian data kepada pelabel eksternal.
3. Memvalidasi hasil pelabelan eksternal untuk memastikan kesesuaian dengan standar kualitas yang ditentukan.
4. Melakukan pra-pemrosesan data agar sesuai dengan format yang dibutuhkan, khususnya format YOLOv7 untuk model deteksi objek.
5. Melatih model AI menggunakan *dataset* yang telah disiapkan, kemudian menganalisis hasil pelatihan untuk mengevaluasi akurasi prediksi dan mengidentifikasi area yang memerlukan perbaikan.
6. Mendokumentasikan seluruh aktivitas secara berkala dalam bentuk laporan internal guna mendukung proses evaluasi dan pelaporan proyek.

Dengan pelaksanaan kegiatan tersebut, kontribusi terhadap peningkatan kualitas dan keandalan sistem berbasis AI dapat terwujud melalui pendekatan yang terstruktur, kolaboratif, dan berbasis evaluasi menyeluruh.

3.3 Uraian Pelaksanaan Magang

Kegiatan magang dilaksanakan secara bertahap dan berkelanjutan, dengan fokus utama pada proses QA dan pengembangan sistem kecerdasan buatan (AI). Setiap minggu memiliki cakupan tugas yang berbeda, mengikuti alur peningkatan kompleksitas dan tanggung jawab teknis yang diberikan. Seluruh aktivitas yang

dilakukan bertujuan untuk mendukung pengembangan sistem berbasis AI yang andal serta memberikan kontribusi nyata dalam proses validasi dan peningkatan performa model. Rincian pelaksanaan kegiatan magang dapat dilihat pada Tabel 3.1 berikut:

Tabel 3.1. Pekerjaan mingguan selama pelaksanaan kerja magang

Minggu Ke -	**Pekerjaan yang dilakukan**
1	Memahami arsitektur sistem perusahaan dan mulai mempelajari serta menerapkan konfigurasi aturan deteksi sebagai dasar untuk *quality assurance* berbasis visual data.
2	Melatih model Roboflow dan menyiapkan *dataset* validasi berbasis waktu, termasuk evaluasi performa model untuk meningkatkan akurasi deteksi melalui penyesuaian terhadap data dan konfigurasi.
3	Menyusun laporan *quality assurance* mingguan, membersihkan data dari deteksi yang salah, dan mengembangkan format laporan *time-series* untuk pelacakan hasil QA yang lebih jelas dan efisien.
4	Menulis skrip Python untuk menggabungkan, menstandarkan, dan membersihkan *dataset* anotasi sebagai persiapan pelatihan model yang lebih terstruktur dan siap digunakan.
5	Melakukan evaluasi performa model berdasarkan data hasil deteksi dan QA, serta mendokumentasikan hasilnya dalam laporan sebagai dasar diskusi dan pengambilan keputusan tim teknis.
6	Mengembangkan proses pembersihan data terstandar untuk berbagai tahap pelatihan model, mengintegrasikan data ke alat anotasi internal, serta melakukan QA berdasarkan aturan model baru guna meningkatkan ketepatan evaluasi performa.
7	Mengembangkan laporan QA harian berbasis *time-series* dengan fokus pada *false positives* dan *edge cases*, serta melakukan pelatihan beberapa versi model untuk membandingkan pengaruh variasi *dataset* terhadap performa.

Lanjut pada halaman berikutnya

Tabel 3.1 Pekerjaan mingguan selama pelaksanaan kerja magang (lanjutan)

Minggu Ke -	**Pekerjaan yang dilakukan**
8	Meningkatkan model melalui *debugging* berbasis data yang ada, dengan cara mengambil deteksi salah (*false positive*), melakukan pelabelan ulang, dan menambahkan hasilnya ke *dataset* untuk menghasilkan versi model yang lebih akurat.
9	Menulis dan menerapkan berbagai skrip *Python* untuk membersihkan dan menstandarkan data, serta memproses dan mengumpulkan *dataset* secara berkala sebagai persiapan pengembangan model berikutnya.
10	Mengevaluasi performa model terbaru secara sistematis untuk mengidentifikasi area yang perlu ditingkatkan dan merumuskan strategi pengembangan model lanjutan.
11	Melakukan pembersihan deteksi pada UI pengguna agar hasil akhir yang diterima klien bebas dari kesalahan deteksi, sekaligus memperkuat validitas statistik deteksi model.
12	Mengembangkan proyek demonstrasi deteksi video untuk *tenant* lain menggunakan YOLOv11, dimulai dari pengumpulan dan pelabelan data, pelatihan model, hingga penyerahan hasil akhir, sambil melakukan *quality assurance* pada data berlabel dari pihak eksternal sebelum digunakan dalam proses pelatihan atau validasi model utama.
13	Menulis panduan pelabelan, mengekstrak data baru, bekerja sama dengan pelabel eksternal untuk melakukan pelabelan data, kemudian melakukan *quality assurance* dan pra-pemrosesan data, serta mengulangi proses ini dalam beberapa siklus data.
14	Melatih model baru untuk deteksi orang dan kendaraan menggunakan *pipeline* YOLOv7 milik tim AI, serta menyiapkan data tambahan untuk pengembangan model lanjutan jika diperlukan.
	Lanjut pada halaman berikutnya

Tabel 3.1 Pekerjaan mingguan selama pelaksanaan kerja magang (lanjutan)

Minggu Ke -	Pekerjaan yang dilakukan
15	Melakukan pemantauan performa model baru dan model lama melalui *quality assurance* berbasis aturan pada tiap kategori deteksi, serta memperbarui *datasheet* sebagai dokumentasi perkembangan performa model.
16	Menyiapkan dan memproses data untuk model deteksi orang dengan menyatukan kelas dan memperbaiki anotasi, melatih model baru menggunakan data yang telah diproses, serta mengkalibrasi tampilan spasial menggunakan Figma dan skrip pendukung.

Pada minggu pertama hingga minggu ketiga, kegiatan difokuskan pada pemahaman arsitektur sistem yang digunakan di perusahaan serta proses *quality assurance* berbasis data visual. Pelatihan model awal dilakukan menggunakan Roboflow dengan penyusunan *dataset* validasi berbasis waktu, diikuti dengan evaluasi performa model. Selain itu, mulai dikembangkan format laporan *quality assurance* mingguan yang memanfaatkan pendekatan *time-series* untuk mempermudah pelacakan hasil deteksi.

Memasuki minggu keempat hingga minggu ketujuh, fokus berpindah pada otomatisasi pengolahan data dan peningkatan struktur *dataset*. Berbagai skrip Python ditulis untuk menggabungkan, menstandarkan, serta membersihkan data anotasi guna mempersiapkan proses pelatihan yang lebih efisien. Model dilatih ulang dengan variasi *dataset*, dan performanya dievaluasi berdasarkan metrik serta deteksi yang relevan. Di saat yang sama, laporan QA harian mulai diterapkan untuk mengamati *false positives* dan kasus batas (*edge cases*), serta digunakan untuk mendukung pengambilan keputusan teknis.

Pada minggu kedelapan hingga minggu kesebelas, dilakukan proses *debugging* model dengan memanfaatkan deteksi yang salah sebagai dasar pelabelan ulang, kemudian hasilnya ditambahkan kembali ke dalam *dataset* untuk pelatihan lanjutan. Evaluasi performa dilakukan secara sistematis dengan menggunakan aturan deteksi, dan hasilnya digunakan untuk memperkuat validitas model. Selanjutnya, integrasi antara proses QA dan antarmuka pengguna (UI) dilakukan agar hasil akhir yang ditampilkan kepada klien telah melewati proses validasi internal secara menyeluruh.

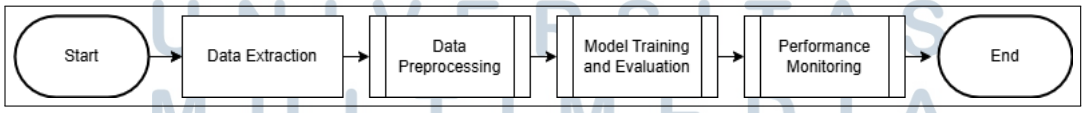
Pada minggu kedua belas, proyek demonstrasi pengujian model baru

dikembangkan secara terpisah sebagai pembuktian konsep untuk kebutuhan *tenant* lain. Proyek ini mencakup seluruh proses dari pengumpulan dan pelabelan data, pelatihan model berbasis YOLOv11, hingga *quality assurance* terhadap data eksternal dan penyampaian hasil akhir. Aktivitas ini menjadi bagian dari simulasi penerapan model dalam konteks berbeda sambil tetap menjaga standar kualitas data.

Pada minggu ketiga belas, kegiatan berfokus pada penyusunan panduan pelabelan dan pengelolaan siklus data baru, dimulai dari ekstraksi data, koordinasi dengan pelabel eksternal, hingga *quality assurance* dan pra-pemrosesan data sebelum digunakan dalam pelatihan. Proses ini dilakukan secara berulang dalam beberapa putaran untuk memastikan konsistensi kualitas data. Minggu keempat belas diisi dengan pelatihan model baru untuk deteksi orang dan kendaraan menggunakan *pipeline* YOLOv7 milik tim AI, serta persiapan data tambahan sebagai antisipasi kebutuhan pengembangan model lebih lanjut.

Pada minggu kelima belas, dilakukan pemantauan performa model baru maupun lama dengan menerapkan *quality assurance* berbasis aturan pada masing-masing kategori deteksi. Selain itu, *datasheet* diperbarui secara berkala sebagai bagian dari dokumentasi perkembangan performa model. Terakhir, pada minggu keenam belas, dilakukan persiapan data tambahan untuk model deteksi orang dengan menyatukan kelas dan memperbaiki anotasi, kemudian melatih model terbaru berdasarkan data tersebut. Di minggu ini pula dilakukan proses kalibrasi spasial menggunakan Figma dan skrip otomatisasi sebagai bagian dari penyempurnaan integrasi visual sistem.

Selain rincian kegiatan mingguan sebagaimana tercantum pada Tabel 3.1, proses kerja magang juga mengikuti alur teknis yang sistematis untuk mendukung peningkatan performa model. Diagram berikut menggambarkan struktur umum alur kerja dalam proyek pengembangan dan validasi model AI selama pelaksanaan magang.



Gambar 3.1. Diagram alur proses pengembangan dan penyempurnaan model

Gambar 3.4 merupakan alur proses *model improvement* yang mencakup seluruh tahapan mulai dari ekstraksi data hingga monitoring performa akhir model. Setiap tahap dirancang untuk memastikan bahwa data yang digunakan dalam pelatihan bersih, terstruktur, dan sesuai dengan standar kualitas yang telah

ditetapkan. Proses ini juga melibatkan validasi berlapis dan pemantauan performa secara berkelanjutan untuk menjaga konsistensi hasil model. Seluruh rangkaian langkah ini bersifat iteratif dan memungkinkan peningkatan model secara bertahap berdasarkan umpan balik.

Tahap pertama adalah *Data Extraction*, di mana data dikumpulkan dari berbagai sumber seperti rekaman video, gambar statis, atau hasil pengambilan data lapangan lainnya. Proses ini bertujuan untuk mengakuisisi data mentah yang relevan dan representatif terhadap kondisi nyata. Data hasil ekstraksi akan menjadi fondasi bagi proses preprocessing selanjutnya. Seleksi dan penyaringan awal juga dilakukan untuk memastikan bahwa data yang akan diproses memiliki kualitas minimum yang dapat diterima.

Tahap berikutnya adalah *Data Preprocessing*, yang mencakup pembersihan data, penyusunan struktur folder, proses sampling, hingga pelabelan dan validasi. Dalam tahap ini, data diberi anotasi melalui metode otomatis maupun manual, kemudian diperiksa ulang oleh tim QA melalui panduan pelabelan yang terstandar. Validasi berlapis dilakukan melalui iterasi label dan pengecekan akhir sebelum data digunakan untuk pelatihan. Selain itu, dilakukan juga penyesuaian resolusi, penggantian gambar terkompresi, dan standarisasi kelas anotasi agar data siap diproses oleh model secara konsisten.

Tahapan *Model Training and Evaluation* dilakukan setelah data selesai diproses dan distandarisasi. Model dilatih menggunakan dataset yang telah dibersihkan, kemudian dievaluasi untuk mengukur performanya dalam mendeteksi objek atau melakukan klasifikasi. Proses ini juga melibatkan tuning hyperparameter dan pengujian terhadap data validasi untuk menghindari *overfitting*. Hasil evaluasi akan menentukan apakah model layak digunakan atau perlu perbaikan pada iterasi berikutnya.

Tahap akhir adalah *Performance Monitoring*, yang berfungsi untuk mengawasi model yang telah diterapkan di lingkungan nyata. Monitoring ini mencakup deteksi performa model dalam konteks operasional serta dokumentasi dan pelaporan hasil secara berkala. Tahap ini penting untuk menangkap anomali yang mungkin terjadi setelah model digunakan secara aktual. Informasi dari tahap ini juga akan menjadi masukan berharga dalam proses *model improvement* pada siklus berikutnya.

3.3.1 Data Extraction

Data yang digunakan dalam proyek ini berada pada server yang telah disiapkan khusus untuk setiap klien atau *tenant*. Pada server tersebut, seluruh hasil *deployment* produk disimpan bersama dengan data-data visual seperti rekaman video. Karena setiap klien memiliki server yang terpisah, proses pengambilan data tidak dapat dilakukan secara langsung dari sistem lokal perusahaan. Oleh karena itu, sebelum pengolahan lebih lanjut dilakukan, data dari server klien perlu dipindahkan ke server internal perusahaan terlebih dahulu. Pemindahan ini menjadi tahap awal yang sangat penting untuk menjamin ketersediaan dan keamanan data selama proses pengolahan.

Proses pemindahan dilakukan menggunakan protokol `SCP` (*Secure Copy Protocol*) yang berbasis `SSH` untuk menjamin keamanan komunikasi data antar server. Dengan menggunakan `SCP`, file dapat ditransfer secara terenkripsi sehingga risiko kebocoran data dapat diminimalkan. Protokol ini memungkinkan transfer data langsung dari server *remote* ke server lokal secara efisien. Dalam konteks ini, `SCP` sangat sesuai karena mendukung pemindahan data berskala besar seperti rekaman video berdurasi panjang. Selain itu, `SCP` juga telah didukung oleh sistem operasi berbasis UNIX/Linux yang umum digunakan di lingkungan server perusahaan.

Cuplikan perintah berikut memperlihatkan bagaimana `SCP` digunakan untuk menyalin file dari server klien ke server lokal perusahaan:

```
scp username@remote_host:/remote/path/to/file /local/path/
```

Kode 3.1: Perintah SCP untuk menyalin data dari server klien

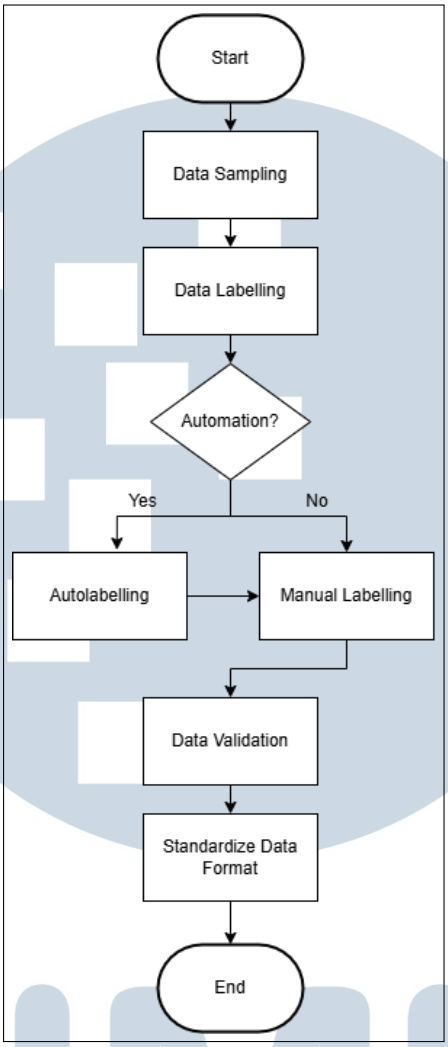
Perintah tersebut dijalankan melalui terminal untuk menginisiasi transfer file dari direktori tertentu di server *remote*. Bagian `username@remote_host` merepresentasikan informasi login ke server klien yang bersangkutan. Path sumber `/remote/path/to/file` mengacu pada lokasi file yang ingin diambil, sedangkan `/local/path/` merupakan direktori tujuan di server perusahaan. Dengan perintah ini, proses transfer dapat dilakukan secara otomatis dan berulang sesuai dengan kebutuhan proyek.

Setelah proses pemindahan selesai, data akan disimpan dalam struktur direktori yang telah ditentukan sebelumnya di server internal perusahaan. Klasifikasi biasanya dilakukan berdasarkan nama klien, tanggal, atau jenis data, sehingga mempermudah pencarian dan pengelolaan. Penyimpanan yang sistematis juga penting untuk menghindari duplikasi serta menjaga versi data yang digunakan

dalam proses selanjutnya. Dengan demikian, tahap ekstraksi ini memberikan fondasi yang kuat sebelum data diproses dalam bentuk *sampling*, anotasi, dan pelatihan model. Prosedur ini juga menjamin bahwa seluruh data yang digunakan telah melalui proses verifikasi awal dari sisi sumber dan struktur penyimpanannya.

3.3.2 Data Preprocessing

Tahapan *data preprocessing* merupakan langkah krusial dalam proses pengembangan model kecerdasan buatan karena kualitas data sangat memengaruhi performa model yang dihasilkan. Melalui proses ini, data yang dikumpulkan dari berbagai sumber akan dipersiapkan secara sistematis agar sesuai dengan format, struktur, dan standar yang dibutuhkan oleh algoritma pembelajaran mesin. Tujuan utama dari preprocessing adalah memastikan bahwa data bebas dari kesalahan, memiliki anotasi yang akurat, dan memiliki konsistensi format untuk memudahkan proses pelatihan. Oleh karena itu, tahapan ini harus dilakukan secara hati-hati agar tidak terjadi bias atau kekeliruan yang dapat memengaruhi hasil akhir model.



Gambar 3.2. Alur proses prapemrosesan data sebelum digunakan dalam pelatihan model.

Gambar 3.2 menunjukkan alur dari proses *data preprocessing* yang dimulai dari kegiatan *data sampling* terhadap data visual yang telah diekstrak sebelumnya. Setelah itu, proses pelabelan dilakukan dengan dua pendekatan, yaitu secara otomatis melalui autolabelling atau secara manual, tergantung dari jenis data dan tingkat keakuratan yang diharapkan. Hasil pelabelan kemudian divalidasi untuk memastikan kesesuaian terhadap standar kualitas yang ditentukan. Terakhir, data yang telah tervalidasi akan distandarkan formatnya, termasuk penyesuaian resolusi, struktur folder, dan nama kelas, sehingga siap digunakan dalam proses pelatihan model AI secara optimal.

A Format dan Struktur Data

Data yang digunakan dalam proyek magang ini merupakan data visual yang disesuaikan dengan format YOLOv7, yaitu sebuah format anotasi yang umum digunakan dalam pelatihan model *object detection*. Setiap data terdiri dari berkas gambar dalam format JPEG atau PNG, serta berkas anotasi pasangan dalam format `.txt`. Berkas anotasi tersebut berisi informasi koordinat dan kelas objek yang terdapat pada gambar. Struktur data seperti ini mempermudah proses pelabelan dan pelatihan karena dapat dibaca langsung oleh model yang digunakan.

Data yang tersedia sebagian besar merupakan cuplikan dari video pengawasan atau proses operasional di lingkungan industri. Oleh karena itu, sebelum digunakan, data berupa video perlu di-*sampling* terlebih dahulu agar dapat dikonversi menjadi gambar-gambar terpisah yang merepresentasikan momen-momen penting dari aktivitas kerja. Proses ini dilakukan secara konsisten untuk memastikan seluruh data yang diproses memiliki keseragaman dalam format dan struktur, sehingga kompatibel dengan *pipeline* pelatihan model.

B Sampling Data

Sampling adalah proses pemilihan bagian tertentu dari keseluruhan populasi data untuk dianalisis lebih lanjut. Dalam konteks ini, populasi yang dimaksud adalah seluruh *frame* dalam rekaman video, sedangkan sampelnya adalah *frame-frame* tertentu yang diambil untuk merepresentasikan konten video secara efisien. Menurut Siyasiya (2024), *sampling* merupakan proses sistematis dalam memilih bagian dari populasi yang diteliti agar hasilnya dapat digeneralisasi secara adil terhadap keseluruhan data yang tersedia [1]. Melalui proses *sampling*, jumlah data dapat dikendalikan tanpa kehilangan keberagaman informasi yang diperlukan untuk pelatihan model deteksi objek. Dengan demikian, proses ini memungkinkan efisiensi pemrosesan dan penggunaan sumber daya komputasi secara lebih optimal.

Dalam praktiknya, *sampling* dilakukan dengan terlebih dahulu memilih rekaman berdasarkan jam atau waktu tertentu yang berpotensi mengandung kejadian penting, seperti rekaman simulasi atau saat terjadinya *false positive* (*FP*). Setelah itu, dilakukan proses ekstraksi *frame* dari video menggunakan skrip otomatisasi berbasis Python dan pustaka OpenCV. Cuplikan kode berikut menunjukkan bagaimana proses ekstraksi *frame* dilakukan:

```
cap = cv2.VideoCapture(video_path)
```

```
fps = int(cap.get(cv2.CAP_PROP_FPS))
interval = int(fps * 60 * sampling_minutes)
current_frame = 0

while cap.isOpened():
    current_frame += interval
    cap.set(cv2.CAP_PROP_POS_FRAMES, current_frame)
    ret, frame = cap.read()
    if not ret:
        break
    frame = cv2.resize(frame, (1920, 1080))
    cv2.imwrite(f"output_dir/frame_{current_frame}.jpg", frame)
```

Kode 3.2: Cuplikan kode untuk sampling frame dari video

Kode tersebut berfungsi untuk mengambil *frame* dari video setiap satu detik (dengan interval dalam satuan menit) kemudian menyimpannya sebagai gambar berukuran standar. Dengan pendekatan ini, proses ekstraksi dari video berdurasi panjang dapat dilakukan secara otomatis dan efisien. Hasil ekstraksi kemudian digunakan sebagai dasar untuk proses anotasi dan pelabelan.

Setelah *frame* berhasil diambil, dilakukan seleksi ulang menggunakan teknik *purposive sampling*, yaitu pemilihan *frame* secara manual berdasarkan tujuan tertentu. Dalam konteks ini, *frame* yang dipilih adalah yang paling representatif terhadap kejadian yang ingin ditangkap, seperti keberadaan objek target atau skenario yang sulit dideteksi. Dengan metode ini, proses pelabelan dapat difokuskan hanya pada data yang bernilai tinggi dalam pelatihan, serta dapat membantu mengurangi terjadinya kesalahan klasifikasi seperti *false positive* maupun *false negative*.

C Pelabelan Data

Pelabelan data dalam proyek ini dilakukan melalui dua pendekatan utama, yaitu secara otomatis dan manual. Pelabelan otomatis, atau yang dikenal dengan istilah *autolabelling*, dilakukan dengan memanfaatkan model deteksi objek yang telah dimiliki sebelumnya untuk menghasilkan anotasi secara otomatis. Sementara itu, pelabelan manual dilakukan secara langsung oleh manusia dengan menggunakan alat bantu anotasi. Kedua metode ini digunakan secara saling melengkapi, di mana pelabelan otomatis digunakan untuk mempercepat proses, sedangkan pelabelan manual digunakan untuk menjamin kualitas dan akurasi hasil

anotasi. Penjelasan lebih rinci mengenai masing-masing pendekatan dijelaskan dalam subbagian berikut.

C.1 Pelabelan Otomatis (Autolabelling)

Pada pendekatan otomatis, proses pelabelan dilakukan dengan menggunakan model deteksi objek awal yang telah tersedia sebelumnya. Untuk mendukung implementasinya, sebuah *pipeline* anotasi internal telah dibangun oleh tim AI/ML agar proses ini dapat dijalankan secara efisien. Dalam pipeline tersebut, gambar hasil *sampling* akan dialirkan ke model untuk diprediksi posisi serta kelas objeknya, kemudian hasilnya akan diekspor ke dalam berkas teks (`.txt`) dengan format anotasi sesuai standar YOLOv7. Setiap gambar akan memiliki pasangan berkas anotasi yang memuat informasi berupa kelas objek, koordinat pusat, serta dimensi *bounding box*. Dengan pendekatan ini, proses pelabelan awal dapat dilakukan secara cepat, terutama untuk kelas-kelas yang telah dikuasai oleh model.

Namun, karena model yang digunakan dalam proses ini merupakan model yang sedang dikembangkan dan diperbaiki, akurasi prediksi yang dihasilkannya belum sepenuhnya dapat diandalkan. Kesalahan anotasi masih sering ditemukan, terutama dalam bentuk *false positive* maupun *false negative* terhadap objek-objek tertentu. Oleh karena itu, hasil dari proses autolabelling tetap harus melalui tahap validasi dan koreksi oleh annotator manusia. Proses peninjauan ulang ini sangat penting untuk memastikan bahwa dataset yang digunakan dalam pelatihan tidak mengandung kesalahan label yang dapat memengaruhi kinerja model secara negatif. Dengan demikian, autolabelling hanya digunakan sebagai langkah akselerasi awal, bukan sebagai pengganti total dari proses anotasi manual.

C.2 Pelabelan Manual

Pelabelan manual dilakukan secara langsung oleh annotator dengan menggunakan platform berbasis web bernama Roboflow. Platform ini menyediakan antarmuka grafis yang memungkinkan pengguna untuk menggambar *bounding box* secara intuitif di atas objek yang terdapat pada gambar. Setiap objek yang terdeteksi harus ditempatkan dalam satu kotak pembatas tersendiri, dan tidak diperkenankan menggabungkan beberapa objek ke dalam satu *bounding box*. Ketelitian dalam menentukan posisi dan ukuran kotak sangat penting, karena hal ini akan memengaruhi kemampuan model dalam mengenali dan membedakan objek

selama proses pelatihan. Selain itu, konsistensi dalam pemberian label juga dijaga agar struktur dataset tetap rapi dan seragam.

Berikut adalah ilustrasi contoh data yang telah dianotasi menggunakan format YOLOv7, di mana setiap objek ditandai dengan *bounding box* berwarna pada gambar:

Gambar 3.3. Contoh format anotasi *YOLOv7*

Setelah proses anotasi dilakukan, berkas pelabelan akan disimpan dalam format teks (`.txt`) dengan struktur standar YOLOv7. Setiap baris pada berkas tersebut merepresentasikan satu objek dalam gambar dan memuat lima komponen data: indeks kelas, posisi tengah objek dalam koordinat ter-normalisasi (x_{center}, y_{center}), serta lebar dan tinggi *bounding box*. Contoh isi dari berkas anotasi adalah sebagai berikut:

```
1 0.2286995 0.582105 0.107623 0.0821052
1 0.86345 0.5044 0.1695964 0.0798107
2 0.4571748 0.1493684 0.2417488 0.0844
0 0.4443497 0.3861263 0.570717 0.3006736
```

Kode 3.3: Contoh isi file anotasi YOLOv7 (.txt)

Selain file anotasi untuk setiap gambar, proses pelabelan juga melibatkan penyusunan file konfigurasi `data.yaml` sebagai bagian dari struktur *dataset*. File ini berfungsi untuk mendeskripsikan direktori lokasi data pelatihan, validasi, dan pengujian, serta menjelaskan jumlah kelas dan nama-nama kelas yang digunakan. File ini sangat penting karena menjadi referensi utama dalam proses pelatihan

model YOLOv7, sehingga model dapat memahami isi dan struktur dataset secara tepat. Contoh isi dari berkas `data.yaml` adalah sebagai berikut:

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 3
names: ['bare_shirt', 'gloves', 'helmet']
```

Kode 3.4: Contoh file data.yaml

Properti `train` dan `val` menunjukkan lokasi data pelatihan dan validasi, sedangkan `nc` merupakan jumlah kelas dalam dataset, dan `names` berisi daftar label sesuai urutan indeksnya. Dengan konfigurasi ini, proses pelatihan dapat dijalankan secara otomatis tanpa perlu menyusun ulang struktur data secara manual. Kombinasi antara anotasi yang tepat dan konfigurasi yang jelas akan memastikan proses pelatihan berjalan secara optimal dan konsisten.

D Validasi Data

Validasi Data merupakan tahap penting dalam memastikan bahwa label yang digunakan untuk pelatihan model telah sesuai dengan standar kualitas yang ditetapkan oleh perusahaan. Proses ini dilakukan oleh *quality assurance engineer* dengan cara meninjau hasil pelabelan yang telah dikerjakan secara manual oleh labeler eksternal. Karena labeler eksternal umumnya belum memahami konteks spesifik proyek dan standar perusahaan, maka diperlukan pedoman yang jelas agar anotasi yang dihasilkan dapat memenuhi kualitas yang diharapkan. Oleh karena itu, setiap labeler diwajibkan mengikuti panduan pelabelan yang telah disusun oleh tim AI/ML bekerja sama dengan tim *quality assurance*, yang akan dijelaskan lebih lanjut dalam subbagian berikut.

D.1 Panduan Pelabelan (Labelling Guide)

Untuk memastikan konsistensi dan akurasi dalam proses anotasi, seluruh labeler eksternal diwajibkan mempelajari dan mengikuti *labelling guide* yang telah dirancang khusus oleh tim AI/ML bersama tim *quality assurance*. Dokumen panduan ini disusun berdasarkan kebutuhan spesifik dari setiap klien atau *tenant* Kinetixpro dan disimpan dalam bentuk dokumen desain interaktif pada platform

Figma. Dengan memisahkan panduan berdasarkan klien, maka setiap proyek dapat memiliki kriteria pelabelan yang sesuai dengan konteks data masing-masing.

Labelling guide ini mencakup instruksi tentang cara menggambar *bounding box* yang benar, batasan-batasan yang harus diperhatikan, serta pengelompokan kelas objek ke dalam kategori tertentu. Kategori tersebut, misalnya, meliputi *Personal-Forklift-Vehicle* (PFV) dan *Personal Protective Equipment* (PPE), di mana masing-masing kategori memiliki pendekatan anotasi yang berbeda. Setiap kelas akan disertai contoh visual berupa objek yang seharusnya diberi label dan objek yang tidak perlu dilabeli. Penjelasan teks yang mendampingi visual tersebut berfungsi untuk menjabarkan kriteria kualitas anotasi yang baik.

Sebagai contoh, jika sebuah objek terlihat buram atau tertutup sebagian, maka labeler diarahkan untuk mengabaikan objek tersebut atau, dalam beberapa kasus, menetapkannya sebagai kelas *unknown*. Tujuan dari instruksi ini adalah untuk menghindari masuknya data yang ambigu atau tidak representatif (*dirty data*) ke dalam proses pelatihan model. Dengan adanya standar ini, kualitas *dataset* dapat dijaga agar tetap bersih, konsisten, dan tidak menurunkan performa model yang akan dilatih.

D.2 Pelabelan Iteratif (Iterative Labelling)

Proses pelabelan dilakukan secara bertahap melalui pendekatan iteratif yang dibagi dalam beberapa gelombang data (*round*) dan kelompok kecil (*batch*). Setiap *round* merepresentasikan satu set data besar yang diberikan untuk proses pelabelan dalam satu siklus kerja, sedangkan *batch* adalah subdivisi dari *round* yang dibagikan kepada masing-masing labeler. Pendekatan ini diterapkan untuk mencegah pencampuran data antar labeler, mempermudah pelacakan progres, serta memastikan efisiensi dalam proses distribusi dan ekstraksi data.

Dengan membagi *dataset* ke dalam *round* dan *batch*, tim AI/ML dan *quality assurance* dapat memonitor kualitas dan konsistensi hasil secara lebih terfokus. Apabila ditemukan kesalahan atau ketidaksesuaian dalam suatu *batch*, maka hanya bagian tersebut yang perlu diperbaiki tanpa memengaruhi keseluruhan data. Selain itu, pendekatan ini juga memungkinkan pemberian umpan balik secara lebih tepat kepada labeler yang bersangkutan sebelum melanjutkan ke *round* berikutnya. Dengan demikian, kualitas anotasi dapat ditingkatkan secara bertahap melalui proses perbaikan yang sistematis dan berkelanjutan.

D.3 Pemeriksaan Kualitas Label (Label QA)

Setelah proses pelabelan selesai dan file hasil anotasi dikembalikan oleh labeler eksternal, tahap berikutnya adalah proses verifikasi oleh *quality assurance engineer*. Pada tahap ini, file hasil label akan diperiksa satu per satu untuk memastikan bahwa tidak ada kesalahan pelabelan yang lolos dari pengawasan sebelumnya. Pemeriksaan ini mencakup validasi teknis, seperti format dan struktur file, serta aspek semantik, seperti kesesuaian label dengan objek pada gambar.

Proses *Label QA* berfungsi sebagai lapisan penyaring tambahan untuk mengeliminasi kemungkinan terjadinya *human error* maupun *technical error* yang mungkin tidak terdeteksi pada tahap sebelumnya. *Quality assurance engineer* akan memverifikasi bahwa setiap label memenuhi standar kualitas yang telah ditentukan dalam *labelling guide* dan tidak terdapat kesalahan sistemik, seperti kesalahan kelas atau anotasi ganda. Dengan adanya tahapan ini, kualitas akhir dari *dataset* anotasi dapat dijaga dengan lebih baik, serta memastikan bahwa data yang akan digunakan untuk pelatihan benar-benar bersih, valid, dan sesuai standar.

E Standarisasi Format Data

Data yang telah selesai dilabel perlu melalui proses standarisasi sebelum digabungkan ke dalam *dataset* besar yang digunakan untuk pelatihan model. Proses ini bertujuan untuk memastikan bahwa seluruh data memiliki format yang seragam, baik dari segi resolusi, kualitas gambar, maupun struktur anotasi. Tanpa proses ini, model yang dilatih berisiko mendapatkan data yang tidak konsisten, sehingga performa akhir dapat terganggu. Standarisasi dilakukan dalam tiga tahap utama, yaitu penyesuaian resolusi, substitusi gambar, dan penyamaan kelas anotasi.

E.1 Penyesuaian Resolusi Gambar

Beberapa data hasil *recording* masih berada pada resolusi tinggi, yaitu 4MP (2560x1440), sementara *dataset* utama menggunakan resolusi 2MP (1920x1080). Maka dari itu, seluruh gambar harus diubah ukurannya agar sesuai dengan standar resolusi model. Selain membantu penyelarasan format, penurunan resolusi juga berfungsi menghemat sumber daya saat pelatihan dan inferensi. Gambar yang hasil penyesuaiannya terlalu kecil akan dihapus secara otomatis agar tidak mengganggu proses pelatihan.

```
frame = cv2.imread(c_img_path)
img_h, img_w, _ = frame.shape
new_img_h, new_img_w = (img_h/original_h)*1080, (img_w/original_w)
    *1920
frame = cv2.resize(frame, (int(new_img_w), int(new_img_h)))
cv2.imwrite(c_img_path, frame)
```

Kode 3.5: Resize otomatis dari 4MP ke 2MP

Cuplikan kode di atas menunjukkan proses perhitungan ulang dimensi gambar berdasarkan proporsi dari resolusi awal. Gambar kemudian diubah ukurannya dengan fungsi `cv2.resize` dan ditulis kembali ke direktori yang sama. Hal ini dilakukan secara otomatis pada seluruh gambar dalam satu direktori dataset, menjamin efisiensi dan konsistensi.

E.2 Substitusi Gambar Terkompresi

Gambar yang diekspor dari Roboflow sering mengalami kompresi otomatis serta perubahan nama berkas menjadi format acak. Untuk mempertahankan kualitas data, dilakukan proses subtitusi gambar dengan versi asli beresolusi tinggi. Karena nama file dari Roboflow mengandung karakter tambahan seperti hash, maka perlu dilakukan normalisasi nama terlebih dahulu agar cocok dengan gambar aslinya. Proses ini sangat penting terutama ketika anotasi sudah melekat pada gambar, sehingga file pengganti harus memiliki nama yang identik.

```
def clean_filename(filename):
    name, ext = os.path.splitext(filename)
    name = re.sub(r"_jpg\.rf\.[a-f0-9]+$", "", name)
    return name + ext
```

Kode 3.6: Penghapusan penanda nama file Roboflow

Fungsi `clean_filename` di atas menghapus bagian nama file yang ditambahkan oleh Roboflow, seperti pola `_jpg.rf.xxxxx`. Dengan proses ini, seluruh nama gambar kembali ke format semula dan dapat dicocokkan dengan file asli yang tidak dikompresi. Substitusi ini dilakukan secara otomatis pada seluruh gambar dan label dengan bantuan pustaka `os` dan `tqdm` untuk memantau progres.

E.3 Standarisasi Kelas Anotasi

Penggabungan banyak dataset dari berbagai proyek berpotensi menimbulkan ketidaksesuaian struktur kelas. Beberapa dataset lama menggunakan daftar kelas

yang berbeda atau urutan yang tidak konsisten. Untuk menghindari konflik pada saat pelatihan, dilakukan proses remapping kelas agar seluruh anotasi sesuai dengan struktur kelas terbaru. Hal ini penting untuk memastikan setiap kelas dikenali secara konsisten oleh model, tanpa duplikasi atau ketidaksesuaian.

```
old_class_id = int(parts[0])
new_class_id = class_map[old_class_id]
parts[0] = str(new_class_id)
```

Kode 3.7: Remapping indeks kelas lama ke standar baru

Pada cuplikan di atas, nilai indeks kelas dari file anotasi diubah dengan indeks baru berdasarkan pemetaan (*mapping*) kelas lama ke kelas standar. Proses ini dilakukan pada seluruh file label dalam struktur direktori dataset, menjamin bahwa tidak ada anotasi yang terabaikan atau salah klasifikasi. Dengan ini, data yang telah distandarisasi siap digunakan dalam proses pelatihan model secara optimal.

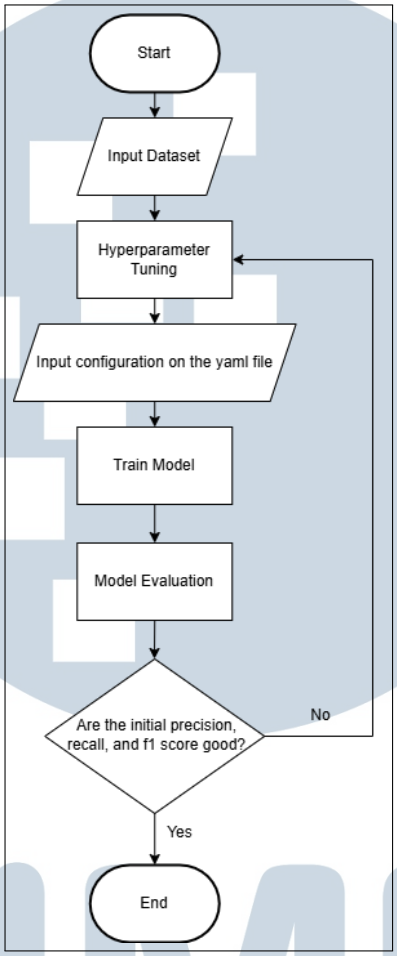
3.3.3 Pelatihan Model dan Evaluasi Awal

Pelatihan model dilakukan sebagai bagian utama dalam membangun sistem deteksi berbasis *You Only Look Once version 7* (YOLOv7), dengan memanfaatkan dataset internal yang telah dibagi menjadi data pelatihan, validasi, dan pengujian. Model dilatih dengan tujuan untuk menghasilkan detektor objek yang akurat dan efisien dalam konteks penerapan AI di lingkungan kerja industri. Algoritma YOLOv7 dipilih karena memiliki arsitektur jaringan yang lebih cepat dan tangguh, serta metode integrasi fitur yang lebih baik. Selain itu, YOLOv7 menawarkan kinerja deteksi objek yang lebih baik, fungsi loss yang lebih kuat, efisiensi pelatihan yang lebih tinggi, dan skema penetapan label yang telah dioptimalkan [2]. Dengan keunggulan tersebut, proses pelatihan dapat dilakukan dengan lebih efektif dan berpotensi menghasilkan model yang siap untuk digunakan pada tahap deployment.

A Proses Pelatihan Model

Sebelum model dapat digunakan untuk mendeteksi objek secara efektif, diperlukan serangkaian tahapan pelatihan yang dilakukan secara sistematis. Seluruh proses pelatihan ini disusun dalam bentuk alur kerja yang bertujuan untuk mengoptimalkan hyperparameter dan arsitektur model agar menghasilkan performa terbaik. Setiap langkah pada proses ini harus dirancang secara hati-hati karena akan berdampak langsung pada hasil deteksi dan generalisasi model. Diagram alur

berikut digunakan untuk menggambarkan tahapan-tahapan penting dalam proses pelatihan dan evaluasi awal model.



Gambar 3.4. Diagram Alur Proses Pelatihan dan Evaluasi Model

Proses pelatihan diawali dengan pengumpulan dan pengorganisasian dataset ke dalam struktur folder proyek model development. Setelah itu, dilakukan proses *hyperparameter tuning* untuk menentukan hyperparameter terbaik dalam pelatihan. Hyperparameter yang diatur mencakup:

1. Confidence Threshold: Nilai ambang batas probabilitas yang menentukan apakah suatu prediksi akan dianggap valid.
2. Batch Size: Jumlah sampel yang diproses dalam satu iterasi, yang memengaruhi kestabilan dan kecepatan pelatihan.
3. Epochs: Jumlah perulangan pelatihan terhadap seluruh dataset, yang berdampak pada tingkat konvergensi model.

4. Inference Resolution: Resolusi input gambar saat inferensi yang memengaruhi detail deteksi dan performa waktu nyata.

Setelah hyperparameter ditentukan, konfigurasi tersebut dicatat dalam sebuah berkas `.yaml` untuk selanjutnya digunakan sebagai referensi saat menjalankan skrip pelatihan model. Pelatihan kemudian dijalankan dengan perintah Python yang akan memanfaatkan konfigurasi tersebut untuk melatih model secara otomatis berdasarkan struktur data dan parameter yang telah ditentukan.

B Evaluasi Hasil Pelatihan

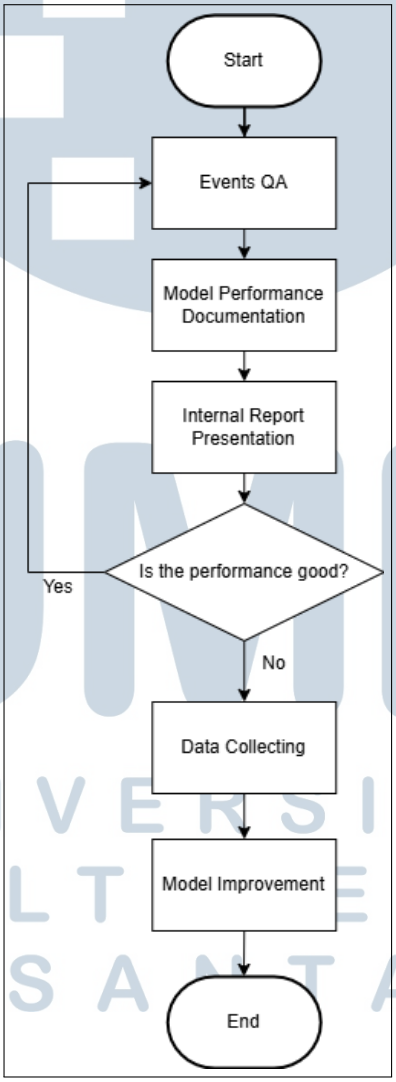
Setelah proses pelatihan selesai dilakukan, langkah selanjutnya adalah melakukan evaluasi terhadap performa awal model. Evaluasi ini penting untuk menentukan apakah model telah mencapai tingkat akurasi dan ketelitian yang layak sebelum diterapkan dalam skenario nyata. Penilaian dilakukan menggunakan metrik-metrik umum seperti *precision*, *recall*, dan *F1-score* yang memberikan gambaran kualitas prediksi model secara kuantitatif. Jika hasil metrik sudah memenuhi kriteria yang ditentukan, maka model dapat dilanjutkan ke tahap penerapan, namun jika belum, perlu dilakukan penyesuaian ulang terhadap data maupun hyperparameter pelatihan.

1. Precision: Mengukur proporsi prediksi positif yang benar, atau seberapa banyak objek yang terdeteksi benar dibandingkan dengan seluruh deteksi positif yang dilakukan oleh model.
2. Recall: Mengukur kemampuan model dalam menemukan semua objek yang relevan dalam data, atau seberapa banyak objek yang benar-benar ada berhasil dideteksi.
3. F1 Score: Merupakan rata-rata harmonis dari precision dan recall, yang memberikan penilaian seimbang terhadap kedua metrik tersebut, terutama ketika distribusi kelas tidak seimbang.

Jika nilai metrik masih rendah, maka perlu dilakukan langkah-langkah lanjutan seperti mengevaluasi ulang kualitas dataset, menyesuaikan hyperparameter pelatihan, atau menambah variasi data melalui augmentasi. Evaluasi ini bersifat iteratif dan akan terus dilakukan sampai model menunjukkan performa yang optimal.

3.3.4 Performance Monitoring

Proses *performance monitoring* dilakukan secara berkelanjutan dengan tujuan untuk mencatat data hasil deteksi dan mengevaluasi performa model dalam konteks operasional aktual. Dengan pemantauan ini, kondisi performa model dapat dinilai secara kuantitatif dan kualitatif untuk menentukan apakah sistem deteksi sudah berjalan secara optimal. Selain itu, hasil evaluasi akan menjadi dasar dalam pengambilan keputusan apakah diperlukan peningkatan model atau cukup dengan pemeliharaan performa saat ini. Proses ini menjadi bagian penting dalam memastikan sistem AI dapat beradaptasi dengan kebutuhan pengguna dan kondisi lapangan yang dinamis.



Gambar 3.5. Flowchart proses pemantauan performa sistem deteksi.

Pada Gambar 3.5, ditampilkan alur proses pemantauan performa sistem deteksi visual. Proses dimulai dari kegiatan QA terhadap *event detection* yang muncul di antarmuka pengguna (UI). Setiap hasil deteksi yang relevan akan dicatat dan didokumentasikan dalam sistem pelaporan berbasis Google Sheets. Setelah itu, data yang telah terdokumentasi akan dipresentasikan dalam rapat internal untuk memperoleh masukan dan evaluasi dari berbagai pihak. Jika hasil evaluasi menunjukkan bahwa performa model sudah mencukupi dan memenuhi kebutuhan operasional, maka sistem dipertahankan sambil terus diawasi. Namun, apabila ditemukan indikasi performa yang menurun atau tidak sesuai ekspektasi, maka proses pengumpulan data tambahan akan dimulai kembali untuk kemudian dilakukan peningkatan model secara menyeluruh.

A Pengecekan Deteksi

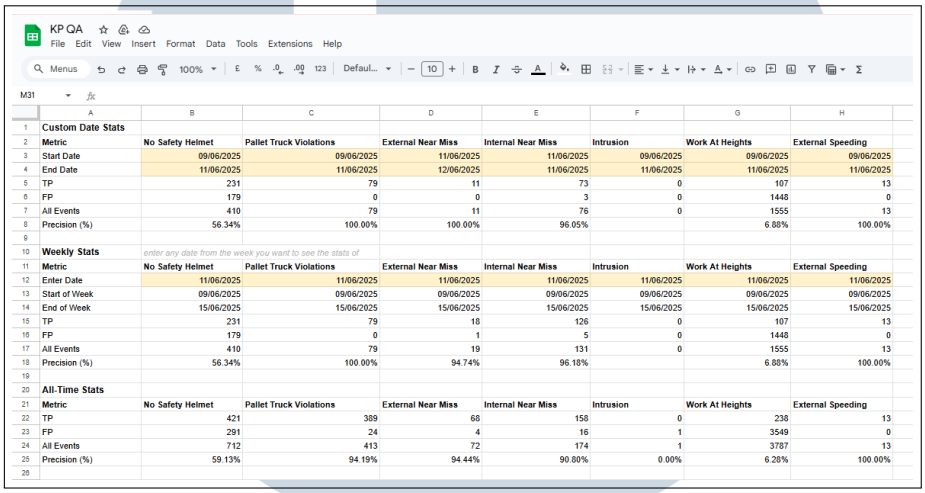
Proses QA terhadap hasil deteksi dilakukan melalui UI web yang disediakan khusus untuk setiap tenant dalam sistem. Pada antarmuka ini, tim Quality Assurance memiliki akses terhadap berbagai fitur bantu yang mempermudah proses evaluasi. Salah satu fitur utama adalah tombol *False Positive*, yang memungkinkan pengguna untuk menandai deteksi yang tidak relevan secara langsung. Selain itu, tersedia juga opsi *flagging* untuk menandai *event* tertentu yang membutuhkan peninjauan lebih lanjut.

Proses QA juga didukung oleh fitur *recall rule*, yang digunakan untuk memverifikasi apakah sistem berhasil mendeteksi objek yang seharusnya terdeteksi berdasarkan aturan yang telah dikonfigurasi. Setiap *rule* dilengkapi dengan konfigurasi spesifik yang mencakup pengaturan area deteksi, pilihan kamera tertentu, serta hyperparameter *custom* sesuai dengan kebutuhan tenant. Dengan adanya konfigurasi ini, sistem dapat dioptimalkan untuk konteks penggunaan yang sangat spesifik, seperti lokasi pabrik tertentu atau jenis pelanggaran keselamatan kerja yang unik. Seluruh hasil dari proses QA ini akan tercermin dalam laporan dan menjadi masukan untuk evaluasi performa model secara menyeluruh.

B Dokumentasi dan Pelaporan Internal Berkala

Selama pelaksanaan proyek, dokumentasi data dilakukan secara sistematis dalam bentuk laporan berkala menggunakan *platform* Google Sheets. Laporan ini dirancang menyerupai sebuah sistem basis data sederhana yang terdiri atas dua

bagian utama, yaitu *page* “Database” dan “Dashboard”. Pada halaman “Database”, setiap entri data dicatat secara manual berdasarkan kategori *rule-based*, misalnya untuk pelanggaran seperti *No Safety Helmet*. Setiap kejadian diklasifikasikan ke dalam kategori seperti *True Positive*, *False Positive*, dan variannya, seperti *False Detection*, *False Person*, *Face Shield*, *Broken Frame*, dan *Duplicate*. Pendataan ini memungkinkan pemantauan kualitas prediksi model secara mendetail.



	A	B	C	D	E	F	G	H
1	**Custom Date Stats**							
2	**Metric**	**No Safety Helmet**	**Pallet Truck Violations**	**External Near Miss**	**Internal Near Miss**	**Intrusion**	**Work At Heights**	**External Speeding**
3	Start Date	09/06/2025	09/06/2025	11/06/2025	11/06/2025	09/06/2025	09/06/2025	09/06/2025
4	End Date	11/06/2025	11/06/2025	12/06/2025	11/06/2025	11/06/2025	11/06/2025	11/06/2025
5	TP	231	79	11	73	0	107	13
6	FP	179	0	0	3	0	1448	0
7	All Events	410	79	11	76	0	1555	13
8	Precision (%)	56.34%	100.00%	100.00%	96.05%		6.88%	100.00%
9								
10	**Weekly Stats**	*enter any date from the week you want to see the stats of*						
11	**Metric**	**No Safety Helmet**	**Pallet Truck Violations**	**External Near Miss**	**Internal Near Miss**	**Intrusion**	**Work At Heights**	**External Speeding**
12	Enter Date	11/06/2025	11/06/2025	11/06/2025	11/06/2025	11/06/2025	11/06/2025	11/06/2025
13	Start of Week	09/06/2025	09/06/2025	09/06/2025	09/06/2025	09/06/2025	09/06/2025	09/06/2025
14	End of Week	15/06/2025	15/06/2025	15/06/2025	15/06/2025	15/06/2025	15/06/2025	15/06/2025
15	TP	231	79	18	126	0	107	13
16	FP	179	0	1	5	0	1448	0
17	All Events	410	79	19	131	0	1555	13
18	Precision (%)	56.34%	100.00%	94.74%	96.18%		6.88%	100.00%
19								
20	**All-Time Stats**							
21	**Metric**	**No Safety Helmet**	**Pallet Truck Violations**	**External Near Miss**	**Internal Near Miss**	**Intrusion**	**Work At Heights**	**External Speeding**
22	TP	421	389	68	158	0	238	13
23	FP	291	24	4	16	1	3549	0
24	All Events	712	413	72	174	1	3787	13
25	Precision (%)	59.13%	94.19%	94.44%	90.80%	0.00%	6.28%	100.00%
26								

Gambar 3.6. Halaman “Dashboard” untuk Dokumentasi Data.

Halaman “Dashboard” seperti ditampilkan pada Gambar 3.6, secara otomatis mengambil data dari halaman “Database” menggunakan rumus dan *pivot table* untuk menghasilkan visualisasi metrik performa. Di antaranya, jumlah *True Positive* dan *False Positive* ditampilkan bersama dengan nilai presisi dari masing-masing kategori aturan. Jika diperlukan, tersedia juga tabel tambahan untuk menghitung metrik *Recall*, dengan tujuan mengidentifikasi kemungkinan *False Negative* yang tidak terdeteksi oleh sistem.

Selain dokumentasi berbasis *spreadsheet*, pelaporan internal juga dilakukan secara berkala melalui pertemuan tim dua kali setiap minggu. Dalam rapat ini, performa model yang sedang dikembangkan dilaporkan, termasuk perubahan signifikan pada metrik presisi dan identifikasi terhadap kesalahan prediksi yang muncul. Setiap anggota tim, termasuk pengembang dan manajer proyek, memberikan umpan balik terhadap hasil evaluasi serta mendiskusikan langkah-langkah perbaikan yang dapat diambil untuk iterasi berikutnya. Proses ini memastikan bahwa pengembangan model berjalan secara adaptif, dengan masukan yang terus diperbaharui berdasarkan kondisi di lapangan dan kebutuhan pengguna internal.

3.4 Kendala dan Solusi yang Ditemukan

Selama proses pelaksanaan proyek, beberapa kendala teknis dan operasional ditemukan dalam berbagai tahapan. Kendala-kendala tersebut berdampak pada performa model maupun kelancaran proses *Quality Assurance* (QA). Oleh karena itu, berbagai pendekatan dan strategi diterapkan sebagai solusi untuk mengatasi permasalahan tersebut secara efektif. Pada bagian ini, kendala dan solusi disajikan dalam dua subbagian terpisah untuk memudahkan pemahaman.

3.4.1 Kendala yang Dihadapi

Selama proyek ini, terdapat beberapa kendala yang ditemukan dalam proses optimalisasi model AI, antara lain:

1. Model yang baru selesai dilatih ditemukan masih menghasilkan deteksi yang kurang akurat karena sejumlah *false positive* dan objek tidak relevan tetap terdeteksi pada data uji meskipun model telah melewati proses pelatihan.
2. Proses QA yang dilakukan melalui UI web seringkali mengalami gangguan akibat performa sistem yang lambat atau *lagging*, sehingga penandaan dan pelabelan event tidak dapat dilakukan dengan lancar terutama saat volume data sedang tinggi.
3. Hasil pelabelan dari labeler eksternal ditemukan tidak konsisten dengan standar yang telah ditetapkan, di mana masih terdapat kesalahan anotasi seperti posisi tidak tepat, kelas yang keliru, atau objek yang tidak diberi label sama sekali.

3.4.2 Solusi yang Diterapkan

Untuk mengatasi kendala-kendala tersebut, berikut adalah solusi yang diterapkan:

1. Visualisasi hasil prediksi awal model dilakukan bersama tim AI sebelum tahap implementasi agar kesalahan seperti *false positive* dapat diidentifikasi lebih awal dan model diuji terlebih dahulu sebelum diterapkan ke lingkungan produksi.

2. Metode *sampling* data diterapkan selama proses QA dengan hanya mengevaluasi data pada jam-jam tertentu yang dianggap representatif seperti *peak hours*, sehingga beban sistem dapat dikurangi tanpa mengorbankan kualitas evaluasi.
3. Validasi data hasil pelabelan eksternal dilakukan secara berulang dan dilengkapi dengan umpan balik mendetail kepada labeler agar kesalahan yang sama tidak terulang serta pemahaman terhadap standar anotasi dapat diselaraskan.