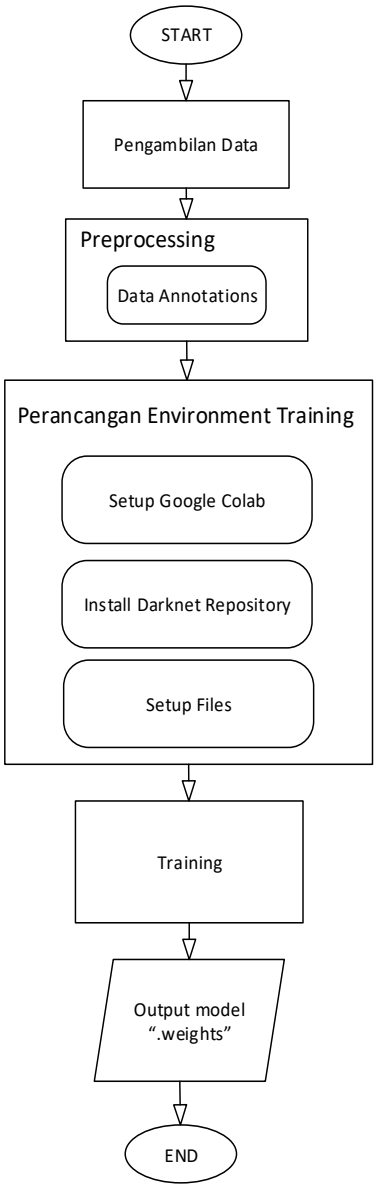
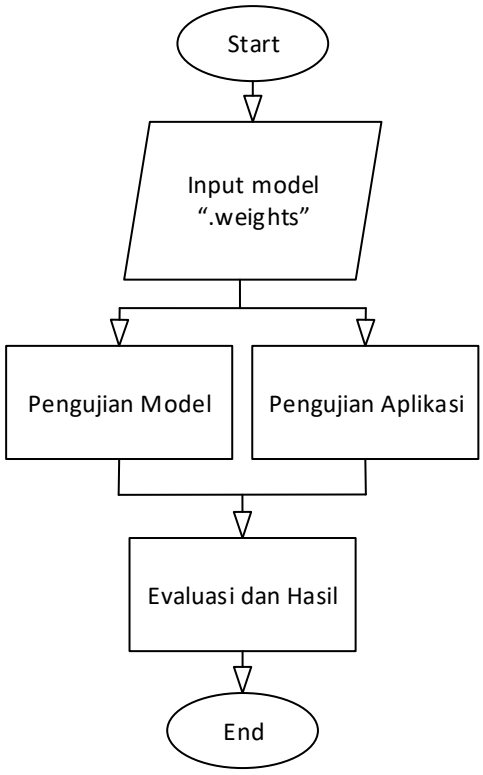
BAB III
ANALISIS DAN PERANCANGAN

3.1 Diagram Alir Penelitian

Sub bab ini membahas tentang urutan langkah-langkah yang dilakukan pada penelitian ini seperti **Gambar 3.1** dan **Gambar 3.2**



Gambar 3.1 Diagram Alir *Training*



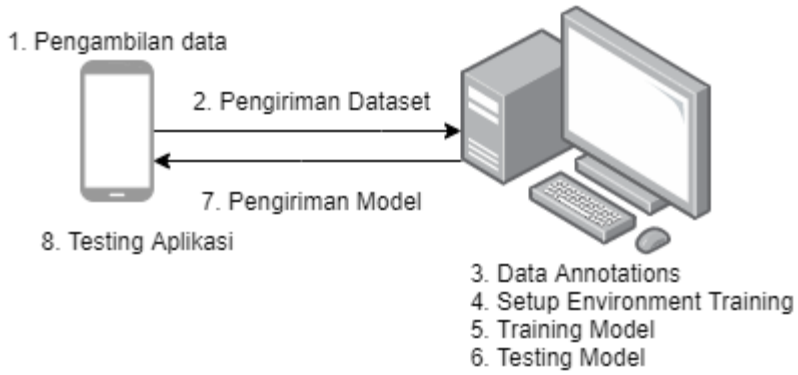
Gambar 3.2 Diagram Alir *Testing*

Berdasarkan kedua gambar di atas, alur penelitian yang dilakukan dimulai dari:

1. Pengambilan data yang digunakan untuk proses pelatihan model.
2. Tahap *preprocessing* data yang sudah dikumpulkan dengan cara menganotasi setiap gambar yang ada.
3. Mempersiapkan seluruh kebutuhan yang diperlukan untuk proses pelatihan. Hal yang dipersiapkan antara lain *Google Colab, darknet repository*, dan konfigurasi pada beberapa *file*.
4. Melakukan pelatihan pada dataset yang ada.
5. Pengujian model yang sudah dilatih.
6. Pengujian aplikasi.
7. Pengambilan kesimpulan dan analisis mengenai penelitian.

3.2 Arsitektur Sistem

Dalam implementasinya, hal-hal yang sudah dijelaskan di atas dijalankan dengan menggunakan dua buah perangkat, yaitu laptop dan sebuah gawai android. Urutan pengerjaan pada masing-masing perangkat dapat dilihat pada **Gambar 3.3.**



Gambar 3.3 Arsitektur Sistem

Penjelasan lebih lanjut mengenai apa saja yang dilakukan pada **Gambar 3.3** terdapat pada sub bab metode penelitian.

3.3 Metode Penelitian

3.3.1 Pengambilan Data

Proses pengambilan data dilakukan dengan cara mengambil gambar pengendara sepeda motor yang menggunakan helm. Total gambar yang diambil adalah 500 gambar. 400 gambar digunakan untuk *training* dan 100 gambar untuk *testing*. Gambar tersebut digunakan untuk proses *training* kelas helm, *person*, dan *motorcycle*. **Gambar 3.4** merupakan salah satu contoh gambar yang digunakan dan mengandung ketiga kelas tersebut.

Gambar 3.4 Sampel Data

3.3.2 *Preprocessing Data Training*

Setiap gambar yang sudah didapatkan pada proses pengambilan data diolah agar dapat digunakan untuk proses *training*. Pada proses ini, setiap gambar diberikan label sesuai dengan setiap kelasnya. Pelabelan dilakukan dengan cara menentukan terlebih dahulu kotak pembatas yang diberikan label. Setelah kotak pembatas ditentukan, akan dipilih kelas apa yang cocok untuk menjelaskan objek yang ada dalam kotak pembatas tersebut. Setelah itu pencatatan kelas dan koordinat kotak pembatas tersimpan dengan format .txt

3.3.3 Perancangan *Environment Training*

Sebelum melakukan proses *training,* harus dilakukan penyesuaian terhadap *environment training* terlebih dahulu. Hal ini dilakukan dengan harapan hasil dari proses *training* yang dilakukan dapat menghasilkan model yang sesuai dengan keinginan. Proses yang dilakukan pada tahap perancangan *environment training* antara lain adalah menghubungkan *Google Colab* dengan *Google Drive*, instalasi *darknet repository*, dan perubahan *file-file* pada *darknet repository* agar disesuaikan dengan kebutuhan *training*.

3.3.4 *Training*

Setelah seluruh data dan konfigurasi sudah siap, maka proses *training* sudah dapat dilakukan. Gambar yang sudah didapatkan pada proses pengambilan data dibagi menjadi dua bagian, yaitu 400 untuk proses *training* dan 120 untuk *testing*. Pembagian data training dan testing dibagi menggunakan metode *split validation* dengan 76.9% data untuk training dan 23.1% untuk testing. Hasil dari proses *training* ini adalah *file* dengan ekstensi *weights* yang dapat digunakan untuk mendeteksi kelas suatu objek pada sebuah gambar maupun video.

3.3.5 Pengujian

Pengujian dilakukan dengan menggunakan 120 gambar. Pada proses pengujian, dicatat berapa jumlah data yang berhasil dideteksi dan gagal. Jika hasil pengujian memiliki akurasi dibawah 50%, maka pelatihan model akan dilakukan kembali sehingga didapatkan model yang sesuai. Pengujian dilakukan dengan menggunakan beberapa gambar dengan kriteria sebagai berikut:

1. Terdapat satu atau lebih pengendara motor yang menggunakan helm pada gambar.
2. Terdapat satu atau lebih pengendara motor yang tidak menggunakan helm pada gambar.
3. Terdapat satu atau lebih pengendara motor yang menggunakan dan tidak menggunakan helm pada gambar.
4. Terdapat orang yang menggunakan helm tanpa motor pada gambar

3.3.6 Evaluasi dan Hasil

Evaluasi dilakukan dengan menggunakan *confusion matrix* dengan mempertimbangkan akurasi, presisi, dan *recall*. Perhitungan akurasi, presisi, dan *recall* dapat dilakukan dengan menggunakan ketiga persamaan di bawah kepada nilai yang didapatkan setelah pengujian model.

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} x100\% \quad \textbf{(3.1)}$$

$$Presisi = \frac{TP}{TP+FP} x100\% \quad \textbf{(3.2)}$$

$$Recall = \frac{TP}{TP+FN} x100\% \quad \textbf{(3.3)}$$

Persentase akurasi, presisi, dan *recall* yang mendekati 100% menggambarkan bahwa model tersebut semakin baik. Dalam mengevaluasi kecepatan algoritma mendeteksi objek dapat dilakukan dengan menampilkan waktu yang dibutuhkan untuk mendeteksi objek tersebut ke layar *handphone*. Waktu yang dicatat adalah dalam satuan milisekon. Semakin sedikit waktu yang dibutuhkan menggambarkan semakin bagus model yang dibuat.

3.4 Diagram Alir Sistem Android

Keseluruhan proses yang dilakukan pada sistem dapat digambarkan dengan diagram alir sistem yang terdapat pada **Gambar 3.5**. Langkah pertama yang dilakukan setelah sistem dimulai adalah meng-*input* gambar yang akan diproses ke dalam sistem. Setelah gambar di-*input*, gambar diubah menjadi matriks agar dapat diproses menggunakan metode YOLO. Setelah gambar selesai diformat, gambar diproses menggunakan metode YOLO. Hasil dari proses menggunakan metode YOLO menghasilkan hasil pendeteksian dengan label yang sudah ditentukan. Label tersebut antara lain adalah *motorcycle*, *person*, dan helm. Kelas-kelas yang sudah berhasil dideteksi akan ditampilkan pada aplikasi dengan diberikan keterangan kelasnya. Label yang didapatkan dari proses pendeteksian juga digunakan dalam proses analisis hasil pendeteksian. Jumlah pengendara motor yang menggunakan helm dan tidak menggunakan helm akan dihitung dalam proses ini. Setelah proses analisis selesai, maka hasil juga ditampilkan pada program.

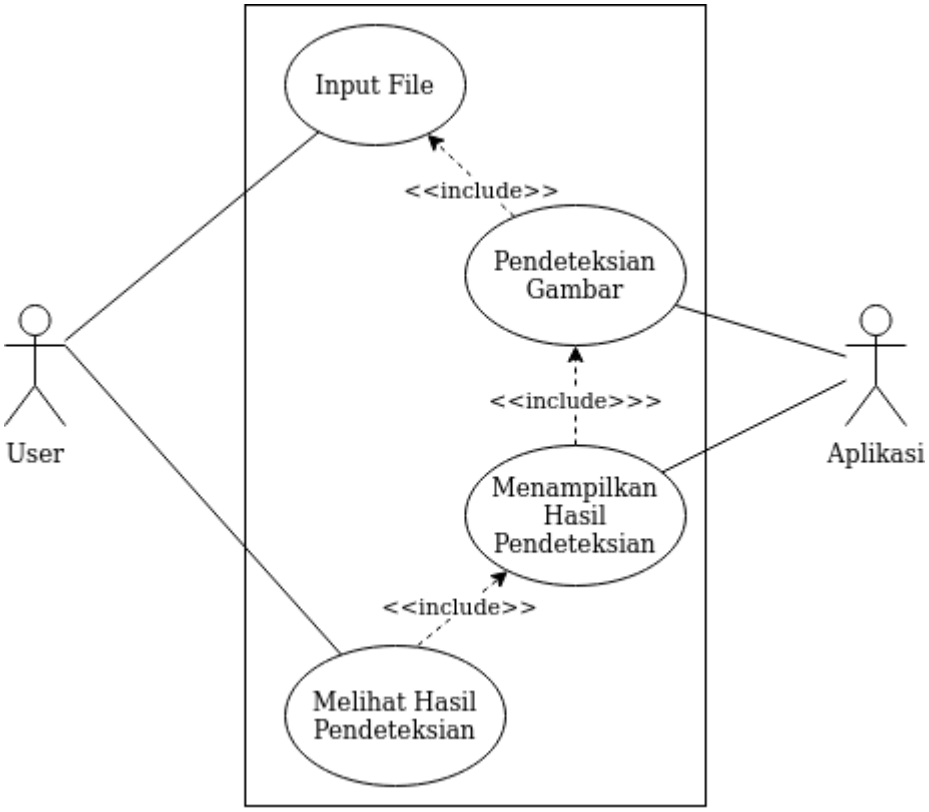
Gambar 3.5 Diagram Alir Sistem

3.4 Rancangan Aplikasi android

3.4.1 *Use Case* Diagram

Use Case merupakan diagram kebutuhan yang menggambarkan fungsionalitas sistem dan aktor-aktornya. Seperti yang digambarkan dalam **Gambar 3.6**, sistem ini memiliki satu pengguna yang dapat meng-*input file* berupa

gambar yang akan diproses. Gambar akan diproses aplikasi, setelah itu akan menampikan gambar yang diproses agar dapat dilihat oleh user.

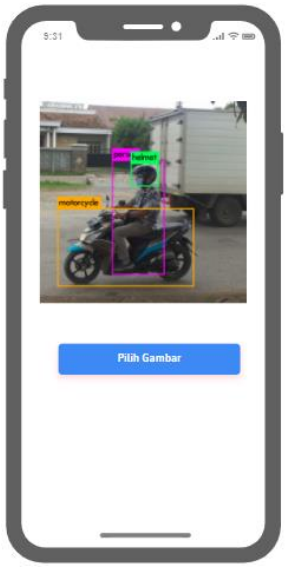


Gambar 3.6 *Use Case Diagram*

3.4.2 Rancangan Antarmuka Sistem

Antarmuka sistem ini memiliki dua komponen. Komponen pertama adalah *image box* yang berfungsi sebagai tempat penampilan gambar pilihan yang sudah diproses. Setelah itu terdapat tombol “Pilih Gambar” yang berfungsi untuk memilih gambar yang akan diproses. Untuk tampilan awal yang lebih jelasnya, dapat dilihat pada **Gambar 3.7.** dan untuk tampilan setelah melakukan pendeteksian dapat dilihat pada **Gambar 3.8.** Pada **Gambar 3.8** terdapat juga keterangan kelas objek yang terdeteksi dengan nilai kepercayaannya.

Gambar 3.7 Antarmuka 1



Gambar 3.8 Antarmuka 2