BAB 5. PENUTUP

5.1. Kesimpulan

Penelitian mengenai Rancang Bangun Sistem Presensi Otomatis Pada Aplikasi Mira melalui pendekatan Software Engineering telah berhasil menyelesaikan rumusan masalah utama. Melalui siklus pengembangan yang terstruktur, sistem ini mampu mengoptimalkan penggunaan sumber daya komputasi serta akurasi pencatatan kehadiran. Integrasi *Human Detection* dan *Face Recognition* terbukti menjadi solusi efektif untuk otomatisasi presensi *real-time* yang akurat dan efisien.

1. Melalui pengembangan software secara iterasi, *agile,* system telah berhasil dirancang menggunakan metode UML yang dapat memenuhi kebutuhan actor serta mengatasi masalah. Proses ini juga membuat solusi pada optimalisasi perangkat lunak dengan menerapkan mekanisme *Labeling System*, *Incremental Variable*, dan *Overlapping* untuk mengatasi kesalahan rekognisi. Penggabungan YOLO dan CNN secara drastis meningkatkan performa: penggunaan CPU/GPU menjadi stabil di kisaran ±25–35%, konsumsi RAM turun ke 55%, dan FPS meningkat tajam dari 3.5 menjadi 27–34 FPS. Implementasi ini juga meningkatkan efisiensi waktu presensi sebesar 61.5% dari 4 menit 20 detik menjadi 1 menit 40 detik. Solusi penempatan kamera di pintu masuk terbukti lebih efektif dibandingkan pemantauan seluruh kelas yang terkendala jarak, didukung dengan skor kepuasan pengguna sebesar 2.53/3.

2. Pencapaian utama lain dari penelitian ini adalah optimalisasi penggunaan sumber daya komputasi. Efisiensi program dievaluasi melalui perbandingan performa model *Face Recognition* (CNN) secara tunggal dan kombinasinya dengan *Human Detection* (YOLO) sebagai *predetector* selama 120 detik. Pengujian menunjukkan bahwa penggunaan CNN secara mandiri menghasilkan konsumsi CPU dan GPU yang fluktuatif, dengan rata-rata 30-40% namun sering melonjak, dan *Frame Per Second* (FPS) yang sangat rendah (sekitar 3.5 FPS). Setelah integrasi dengan YOLO, terjadi peningkatan performa yang drastis. YOLO berhasil membatasi area kerja CNN hanya pada wajah yang terdeteksi, mengurangi volume komputasi secara signifikan. Hasilnya, penggunaan CPU dan GPU menjadi lebih stabil di kisaran ±25% dan ±25–35%, konsumsi RAM turun menjadi sekitar 55%, dan FPS meningkat tajam menjadi rentang 27–34 FPS dengan fluktuasi yang jauh lebih stabil. Secara keseluruhan, integrasi YOLO dan CNN terbukti menghasilkan

performa yang jauh lebih efisien dan stabil, menjadikannya sangat sesuai untuk kebutuhan presensi *real-time*.

3. Meskipun performa sistem secara fungsional sudah baik dan efisien, hasil evaluasi model mengkonfirmasi akurasi yang memadai namun masih memiliki ruang perbaikan. Secara statistik, model *Face Recognition* mencapai akurasi keseluruhan sebesar 87.61% pada 218 sampel uji. *Confusion Matrix* memperlihatkan kemampuan identifikasi yang dominan pada diagonal, tetapi masih terdapat kegagalan pada pelabelan *Unknown*, menunjukkan tantangan dalam membedakan data yang sangat mirip atau belum terdaftar. Meskipun rata-rata *Precision*, *Recall*, dan F1-score per individu berada di atas 0.9, menunjukkan keandalan yang tinggi dalam mengenali sebagian besar individu, model masih dapat ditingkatkan dari segi kecepatan inferensi dimana disini mencapai 6489 ms dan penanganan *outlier* kelas. Di sisi lain, model *Human Detection* (YOLO) menunjukkan performa yang sangat kuat, dengan *Detection Rate* 100.0% dan rata-rata *Confidence* 82.05% serta latensi yang sangat cepat pada angka 46 ms. Secara keseluruhan, akurasi model sudah cukup baik untuk tahap pengujian awal, dan model YOLO sudah sangat efisien dari sisi waktu, namun peningkatan akurasi *Face Recognition* dan efisiensi waktu proses keduanya secara simultan masih dapat dimaksimalkan agar lebih layak digunakan pada aplikasi skala nyata.

5.2. Saran

Berdasarkan hasil penelitian dan evaluasi sistem, terdapat beberapa saran untuk pengembangan dan implementasi lebih lanjut:

1. Peningkatan Akurasi Model *Face Recognition*: Melakukan optimasi pada model *Face Recognition*, seperti penambahan variasi data latih (terutama untuk kasus yang menghasilkan prediksi *Unknown* atau *outlier*), untuk meningkatkan akurasi keseluruhan di atas 90%.

2. Pemanfaatan Akselerasi Perangkat Keras: Mengintegrasikan *framework* akselerasi seperti TensorRT atau *hardware* spesifik (misalnya, *Neural Processing Unit*) untuk mengurangi latensi inferensi *Face Recognition* secara signifikan, yang saat ini masih berada di kisaran ribuan milidetik, agar sistem dapat beroperasi secara lebih *real-time* dan responsif.

3. Pengembangan Skema *Multi-Camera* Adaptif: Untuk solusi presensi kelas secara keseluruhan, perlu dikembangkan skema yang

secara otomatis menggabungkan input dari beberapa kamera beresolusi tinggi, atau menggunakan teknik *image stitching* dan *tracking* yang lebih efektif untuk mengatasi masalah jangkauan deteksi pada barisan belakang.