# BAB V
# PENUTUP

## 5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan terkait penerapan *clean architecture* pada aplikasi berbasis *Flutter* dan dampaknya terhadap *maintanability* dan *complexity* kode, dapat diambil kesimpulan sebagai jawaban dari rumusan masalah, yaitu:

1. Penerapan *clean architecture* dalam pengembangan aplikasi *mobile* berbasis *Flutter* telah berhasil diimplementasikan dengan mengikuti skema *clean architecture* yang terdiri dari tiga lapisan utama, yaitu lapisan presentation, lapisan *domain*, dan lapisan *data*. Setiap lapisan memiliki tanggung jawab dan peran yang berbeda dalam arsitektur aplikasi.
2. Tingkat *maintainability* aplikasi dievaluasi menggunakan *Dart Code Metric*, dan hasilnya menunjukkan nilai *maintainability* index sebesar 78. Nilai ini mencerminkan tingkat *maintainability* yang baik. Analisis lebih lanjut menunjukkan bahwa lapisan *Domain* memiliki tingkat *maintainability* tertinggi (85), diikuti oleh lapisan *Data* (80) dan lapisan Presentation (70).
3. Kompleksitas kode diukur menggunakan *Dart Code Metric*, dengan menghitung nilai *Cyclomatic Complexity*. Hasil evaluasi menunjukkan nilai *Cyclomatic Complexity* keseluruhan sebesar 569. Analisis lebih lanjut menunjukkan bahwa lapisan *Domain* memiliki kompleksitas yang paling rendah (rata-rata 6), sementara lapisan *Data* memiliki kompleksitas yang lebih tinggi (rata-rata 9) dan lapisan Presentation memiliki variasi yang lebih besar dalam kompleksitas (rentang 4-15).

## 5.2 Saran

Sebagai saran, pengembang selanjutnya disarankan untuk menerapkan desain pola yang tepat dan menyederhanakan logika bisnis dalam mengimplementasikan *clean architecture* pada aplikasi berbasis *Flutter*. Pemilihan desain pola yang sesuai dengan kebutuhan aplikasi akan meningkatkan struktur

kode dan memudahkan proses pengembangan. Sementara itu, penyederhanaan logika bisnis dan meminimalisir penggunaan logika yang berulang akan meningkatkan kualitas kode dan memudahkan proses pemeliharaan di masa depan. Selain itu, penting untuk memahami dan menerapkan *BLoC (Business Logic Component)* sebagai salah satu pola desain yang populer dalam pengembangan aplikasi *Flutter*. *BLoC* memungkinkan pengelolaan *state* yang lebih baik dan memisahkan logika bisnis dari antarmuka pengguna, namun memiliki struktur yang lebih kompeleks sehingga di perlukan pemahaman lebih lanjut agar sehingga mendukung prinsip *clean architecture*.