Technical Documentation: fmb

# 1. Overview

fmb adalah aplikasi Flutter multiplatform (Windows, Linux, macOS, Web, Android, iOS) dengan arsitektur modular, state management BLoC, dan dependency injection.

# 2. Architecture Detail

## Modular Structure

- lib/src/features/  
 - Fitur aplikasi: base, home, installation\_wizard, login, welcome\_back  
 - Tiap fitur terdiri dari:   
 - view/ (UI)  
 - bloc/ (state management)  
 - usecase/ (business logic)  
- lib/src/core/  
 - Reusable core:   
 - repositories/ (repository pattern untuk akses data)  
 - services/ (network, database, logger)  
 - di/ (dependency injection/service locator)  
- lib/src/utils/  
 - Utilitas: asset, environment, theme, helper  
- lib/src/models/  
 - Model data  
- assets/  
 - images/, translations/  
- configs/env/  
 - Konfigurasi environment (staging, production)

## State Management

- Menggunakan BLoC (flutter\_bloc) untuk memisahkan UI dan business logic.  
- Setiap fitur memiliki BLoC, State, dan Event sendiri.  
- State diubah melalui event dan dipantau oleh UI menggunakan BlocBuilder/BlocListener.

## Dependency Injection

- Menggunakan get\_it dan injectable.  
- Semua repository, usecase, dan service didaftarkan di service locator (lib/src/core/di/service\_locator.dart).  
- Inisialisasi dependency dilakukan di awal aplikasi (main.dart).

## Repository Pattern

- Setiap akses data (local/remote) menggunakan repository.  
- Contoh: LocalRepository, LoginRepository, DeviceManagementRepository.  
- Memudahkan testing dan pemisahan data source.

## Multi-Platform Support

- Mendukung Windows, Linux, macOS, Web, Android, iOS.  
- Integrasi dengan plugin platform seperti battery, window manager, desktop window, universal\_platform.

## Localization

- Menggunakan easy\_localization.  
- File terjemahan di assets/translations/.  
- Kunci terjemahan di-generate otomatis ke lib/src/utils/locale\_keys.g.dart.

## Routing

Menggunakan go\_router untuk navigasi antar halaman.

## Testing

- Unit test untuk repository, service, dan utilitas di folder test/.  
- Menggunakan mockito untuk mocking dependency.

# 3. Dependencies

Lihat pubspec.yaml untuk detail dependencies utama: flutter\_bloc, easy\_localization, go\_router, dio, get\_it, injectable, shared\_preferences, mockito, dll.

# 4. Setup & Build

- Install: flutter pub get  
- Build & Run: flutter run -d <platform>  
- Jalankan test: flutter test  
- Build runner: dart run build\_runner build --delete-conflicting-outputs  
- Generate locale: dart run bin/generate.dart -S assets/translations -f keys -O lib/src/utils -o locale\_keys.g.dart

# 5. Environment

- File konfigurasi environment di configs/env/ (misal: env.staging.json, env.json).  
- Untuk menjalankan dengan environment tertentu, gunakan argumen:  
 flutter run --dart-define-from-file=configs/env/env.staging.json

# 6. Folder Structure

lib/  
 main.dart  
 src/  
 configs/  
 core/  
 features/  
 models/  
 utils/  
assets/  
 images/  
 translations/  
configs/  
 env/  
test/  
 core/  
 src/

# 7. Features

- Installation Wizard: Proses instalasi awal perangkat.  
- Login: Autentikasi user.  
- Home: Halaman utama aplikasi.  
- Welcome Back: Halaman selamat datang kembali.  
- Multi-language: Mendukung multi bahasa (EN, ID).  
- State Management: Menggunakan BLoC.  
- Dependency Injection: Menggunakan get\_it & injectable.  
- Platform Integration: Battery, window, platform info, dsb.

# 8. Build Runner & Locale

- Untuk generate file otomatis:  
 dart run build\_runner build --delete-conflicting-outputs  
 dart run bin/generate.dart -S assets/translations -f keys -O lib/src/utils -o locale\_keys.g.dart

# 9. Contribution

- Ikuti standar coding Flutter & Dart.  
- Gunakan branch feature/bugfix sesuai kebutuhan.  
- Pastikan semua test lulus sebelum merge.