Gate Pass Management System - Flutter App

A beautiful, feature-rich Flutter application for managing gate passes in educational institutions, built with Material Design 3 and a vibrant yellow theme.

Features

- **Beautiful UI**: Modern design with custom yellow theme (#FFB703)
- **Role-based Access**: Separate dashboards for Students, Teachers, Admins, and Security
- **QR Code Integration**: Generate and scan QR codes for gate passes
- **Real-time Updates**: Live data synchronization with backend
- **Smooth Animations**: Engaging user experience with custom animations
- **Responsive Design**: Works great on all screen sizes

Architecture

- **State Management**: Provider pattern for clean state management
- **Navigation**: GoRouter for type-safe navigation
- **API Integration**: HTTP client with proper error handling
- **Local Storage**: SharedPreferences for authentication persistence
- **Camera Integration**: QR code scanning capabilities

Getting Started

Installation

1. **Clone and Setup**

```
# Create new Flutter project
flutter create gate_pass_flutter
cd gate_pass_flutter

# Replace pubspec.yaml with the provided one
# Copy all the provided Dart files to their respective locations
```

2. **Update Dependencies**

```
flutter pub get
```

3. **Configure API Endpoint**

   Update `lib/services/api_service.dart` with your backend URL:

```
// For local development
static const String baseUrl = 'http://localhost:3001/api';

// For Android emulator
static const String baseUrl = 'http://10.0.2.2:3001/api';

// For physical device (replace with your computer's IP)
static const String baseUrl = 'http://192.168.1.XXX:3001/api';
```

4. **Add Permissions** (Android)

   Add to `android/app/src/main/AndroidManifest.xml`:

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.INTERNET" />
```

5. **Run the App**

```
flutter run
```

6. **API Connection Failed**
   - Check backend is running on correct port
   - Verify API base URL in `api_service.dart`
   - For Android emulator, use `10.0.2.2` instead of `localhost`
7. **QR Scanner Not Working**
   - Ensure camera permissions are granted
   - Test on physical device (emulator camera may not work)
   - Check `qr_code_scanner` package compatibility
8. **Build Errors**
   - Run `flutter clean && flutter pub get`
   - Check Flutter and Dart SDK versions
   - Verify all dependencies are compatible

Performance Tips

- Use `const` constructors where possible
- Implement proper disposal in stateful widgets
- Optimize image loading with `cached_network_image`
- Use `ListView.builder` for large lists

For setup issues or questions about the Flutter app, please refer to:

- Flutter documentation: https://docs.flutter.dev
- Provider documentation: https://pub.dev/packages/provider
- Your backend API documentation