MediNavi - Medical Services Locator & Reminder App

Project Overview

MediNavi is a comprehensive Flutter-based mobile application designed to help users in Japan find nearby medical services and manage their medicine reminders. The app combines location-based healthcare facility search with an intelligent medication management system.

Version Information

- **Version**: 1.0.0+1
- **SDK**: Dart >=3.8.0 <4.0.0
- **Platforms**: iOS, Android
- **Last Updated**: January 2026

Table of Contents

1. [Core Features](#core-features)
2. [Architecture Overview](#architecture-overview)
3. [Project Structure](#project-structure)
4. [Key Components](#key-components)
5. [Services](#services)
6. [Screens](#screens)
7. [Notification System](#notification-system)
8. [Localization](#localization)
9. [Dependencies](#dependencies)
10. [Setup & Installation](#setup--installation)
11. [Configuration](#configuration)
12. [Usage Guide](#usage-guide)
13. [API Integration](#api-integration)
14. [Testing](#testing)
15. [Future Enhancements](#future-enhancements)

Core Features

1. Medical Service Finder

- **Location-Based Search**: Find nearby hospitals, clinics, pharmacies, and specialized medical services
- **Google Maps Integration**: View facilities on interactive maps with real-time directions
- **Category Filtering**: Browse services by categories (General Medicine, Specialized Services, Emergency, etc.)
- **Favorites System**: Save frequently visited medical facilities
- **Multi-Language Support**: Japanese, English, and Chinese interfaces

2. Medicine Reminder System

- **Smart Reminders**: Schedule medicine reminders with customizable frequencies
- One-time reminders
- Daily reminders
- Weekdays only
- Weekends only
- Custom day selection
- **Meal Timing**: Set reminders for before/after meal medications
- **Photo Attachments**: Add photos of medicines for easy identification
- **Push Notifications**: Receive notifications when it's time to take medicine
- **Alarm System**: In-app alarm with audio playback
- **Automatic Tracking**: Mark medicines as taken and track adherence

3. AI Medical Assistant (Chatbot)

- **Voice & Text Input**: Communicate via speech or typing
- **Symptom Analysis**: Describe symptoms to get preliminary guidance
- **Quick & Detailed Modes**: Choose response depth based on your needs
- **Department Recommendations**: Get suggestions for appropriate medical departments
- **Facility Search Integration**: Directly search for recommended services

4. User Authentication

- **Firebase Authentication**: Secure login/signup with email
- **Session Management**: Persistent login state
- **Multi-language Onboarding**: First-time user experience in multiple languages

Architecture Overview

Design Pattern

MediNavi follows a **Service-Oriented Architecture** with clear separation of concerns:

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■              Presentation Layer                   ■
■  (Screens, Widgets, UI Components)               ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
                  ■
■■■■■■■■■■■■■■■■■■■▼■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■              Business Logic Layer                 ■
■  (Services, State Management, Utils)             ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
                  ■
■■■■■■■■■■■■■■■■■■■▼■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■                Data Layer                         ■
■  (Models, Local Storage, API Integration)        ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
```

Technology Stack

• **Framework**: Flutter 3.8.0+

• **Language**: Dart

• **State Management**: Provider pattern with StatefulWidgets

• **Backend**: Firebase (Authentication)

• **Local Storage**: SharedPreferences

• **Notifications**: flutter_local_notifications

• **Maps**: Google Maps API

• **Location**: Geolocator

• **Audio**: audioplayers

Project Structure

```
lib/
■■■ data/
■   ■■■ medical_services_data.dart       # Medical service categories & data
■■■ l10n/                                 # Localization files
■   ■■■ app_en.arb                       # English translations
■   ■■■ app_ja.arb                       # Japanese translations
■   ■■■ app_zh.arb                       # Chinese translations
■   ■■■ app_localizations*.dart          # Generated localization classes
■■■ models/
■   ■■■ medical_service.dart             # Medical service data model
■   ■■■ reminder.dart                    # Medicine reminder model
■■■ providers/
■   ■■■ locale_providers.dart            # Locale state management
■■■ screens/
■   ■■■ home_screen.dart                 # Main home screen
■   ■■■ chatbot_screen.dart              # AI assistant interface
■   ■■■ favorites_screen.dart            # Saved facilities
■   ■■■ map_screen.dart                  # Map view with markers
■   ■■■ settings_screen.dart             # App settings
■   ■■■ startup_screen.dart              # Language selection
■   ■■■ sign_up_screen.dart              # Authentication
■   ■■■ reminder/                        # Medicine reminder screens
■       ■■■ medicine_reminder_screen.dart    # Reminder list
■       ■■■ add_reminder_screen.dart         # Add/edit reminder
■       ■■■ notification_screen.dart         # Alarm notification UI
■■■ services/
■   ■■■ notification_service.dart        # Notification management
■   ■■■ background_alarm_service.dart    # Background notifications
■   ■■■ foreground_alarm_monitor.dart    # Foreground alarm detection
■   ■■■ google_places_service.dart       # Google Places API
■   ■■■ favorites_service.dart           # Favorites persistence
■   ■■■ auth_services.dart               # Firebase auth wrapper
■■■ utils/
■   ■■■ service_localization.dart        # Service name translations
■   ■■■ prefecture_localization.dart     # Location translations
■   ■■■ language_selection_utils.dart    # Language utilities
■■■ widgets/
■   ■■■ reminder_card.dart               # Reminder list item
■   ■■■ service_card_widget.dart         # Service list item
■   ■■■ category_tabs_widget.dart        # Category filter tabs
■   ■■■ location_selector_screen_widget.dart  # Location picker
■■■ main.dart                            # App entry point
```

Key Components

Models

#### Reminder Model

```
class Reminder {
  final String id;
  final String medicineName;
  final String time;
  final String repeatType;      // 'never', 'everyday', 'weekdays', etc.
  final List<int>? customDays;  // Custom day selection
  final String? photoPath;      // Medicine photo
  final String mealTiming;      // 'none', 'before', 'after'
  final int dosesPerDay;        // Number of daily doses
  final List<String> doseTimes; // Times for each dose
  List<bool> takenToday;        // Tracking adherence
  bool isEnabled;               // Active/inactive state
}
```

Key Features:

- Flexible repeat patterns (daily, weekdays, weekends, custom)

- Meal timing support for medication compliance

- Multiple doses per day

- Photo attachment for visual identification

- Daily adherence tracking

#### Medical Service Model

```
class MedicalService {
  final String id;
  final String name;
  final String category;
  final String serviceType;
  final double latitude;
  final double longitude;
  final String address;
  final String? phone;
  final String? website;
  final bool isOpen;
}
```

Services

1. NotificationService

File: `lib/services/notification_service.dart`

Purpose: Central notification management system

Key Methods:

• `initialize()`: Setup notification channels and permissions

• `scheduleNotification()`: Schedule one-time notifications

• `scheduleDailyNotification()`: Schedule recurring daily notifications

• `scheduleReminderDoses()`: Schedule multiple doses for a reminder

• `showAlarmSetConfirmation()`: Show confirmation when alarm is set

• `playAlarm()`: Play alarm sound with looping

• `stopAlarm()`: Stop alarm playback

• `cancelAll()`: Cancel all pending notifications

Platform Support:

• **Android**: High priority notifications with full-screen intent

• **iOS**: Time-sensitive notifications with sound

2. BackgroundAlarmService

File: `lib/services/background_alarm_service.dart`

Purpose: Handle notifications when app is closed or in background

Key Features:

• Schedules local notifications that fire even when app is terminated

• Supports all repeat types (everyday, weekdays, weekends, custom, never)

• Automatically calculates next occurrence times

• Manages notification lifecycle

How It Works:

```
User sets reminder → BackgroundAlarmService schedules notification
                  ↓
       System time reaches scheduled time
                  ↓
    Notification appears in device tray
                  ↓
       User taps notification
                  ↓
        App opens to NotificationScreen
```

3. ForegroundAlarmMonitor

File: `lib/services/foreground_alarm_monitor.dart`

Purpose: Auto-trigger alarm screen when app is in foreground

Key Features:

- Checks every second for matching reminder times
- Automatically opens notification screen
- Plays alarm sound
- Prevents duplicate triggers on same day
- Resets triggers at midnight

Implementation:

```
// Monitors reminders every second
Timer.periodic(const Duration(seconds: 1), (_) {
  // Check if current time matches any reminder
  // If match found and not triggered today:
  //   - Play alarm
  //   - Navigate to notification screen
  //   - Mark as triggered for today
});
```

4. GooglePlacesService

File: `lib/services/google_places_service.dart`

Purpose: Search for medical facilities using Google Places API

Key Methods:

- `searchNearbyPlaces()`: Find facilities near a location
- `getPlaceDetails()`: Get detailed information about a place
- `searchByType()`: Filter by medical service type

API Integration:

- Uses Google Places API (Nearby Search)
- Filters by healthcare-related types
- Returns distance calculations
- Caches API responses

5. FavoritesService

File: `lib/services/favorites_service.dart`

Purpose: Persist user's favorite medical facilities

Storage: SharedPreferences (local JSON storage)

Screens

Home Screen

File: `lib/screens/home_screen.dart`

Features:

- Location-based service discovery
- Category filtering (10+ categories)
- Search by prefecture, city, ward
- Current location detection
- Distance calculation and sorting
- Direct call and directions to facilities

UI Components:

- Location selector dropdown
- Category tabs
- Service cards with distance
- Map view button
- Favorites toggle

Medicine Reminder Screen

File: `lib/screens/reminder/medicine_reminder_screen.dart`

Features:

- List of all medicine reminders
- Enable/disable toggle for each reminder
- Edit and delete actions
- Swipe to delete
- Test alarm functionality
- Empty state with call-to-action

Data Persistence:

- Stores reminders in SharedPreferences as JSON
- Auto-saves on any change
- Loads on app start

Add Reminder Screen

File: `lib/screens/reminder/add_reminder_screen.dart`

Features:

- Medicine name input
- Time picker (12-hour format)
- Repeat options (never, daily, weekdays, weekends, custom)
- Meal timing selection (none, before meal, after meal)
- Photo capture/selection
- Form validation

Repeat Patterns:

1. **Never**: One-time reminder

2. **Everyday**: Repeats every day at same time

3. **Weekdays**: Monday-Friday only

4. **Weekends**: Saturday-Sunday only

5. **Custom**: Select specific days of week

Notification Screen

File: `lib/screens/reminder/notification_screen.dart`

Features:

- Full-screen alarm display
- Medicine photo (if available)
- Time and dose information
- Meal timing badge
- "I took it" button
- Close button
- Auto-dismiss alarm on action

Special Handling:

- After-meal medicines show confirmation dialog
- "Have you eaten?" prompt
- Snooze suggestion for not-yet-eaten

Chatbot Screen

File: `lib/screens/chatbot_screen.dart`

Features:

- Voice and text input
- Quick/Detailed mode toggle
- Chat history
- Department recommendations
- Direct search from chat
- TTS support
- Speech recognition

AI Integration:

- Connects to medical AI API
- Analyzes symptoms
- Suggests appropriate medical departments
- Provides health guidance

Map Screen

File: `lib/screens/map_screen.dart`

Features:

- Interactive Google Maps
- Custom markers for facilities
- Info windows with details
- Directions integration
- Current location marker
- Zoom controls

Favorites Screen

File: `lib/screens/favorites_screen.dart`

Features:

- Saved facility list
- Distance from current location
- Quick call and directions
- Remove from favorites
- Empty state handling

Notification System

Architecture

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■          User Sets Reminder                  ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
               ■
               ▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■   MedicineReminderScreen                     ■
■   - Saves reminder to SharedPreferences      ■
■   - Caches in NotificationService            ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
               ■
       ■■■■■■■■■■■■■■■■
       ■               ■
       ▼               ▼
■■■■■■■■■■■■■■■■  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■ Background   ■  ■ Foreground                ■
■ Service      ■  ■ Monitor                   ■
■              ■  ■                           ■
■ Schedules    ■  ■ Checks every second       ■
■ system       ■  ■ when app is open          ■
■ notifications■  ■                           ■
■■■■■■■■■■■■■■■■  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■
       ■                   ■
       ▼                   ▼
■■■■■■■■■■■■■■■■  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■ Notification ■  ■ Auto-open                 ■
■ appears in   ■  ■ NotificationScreen        ■
■ tray         ■  ■ + Play alarm              ■
```

```
■■■■■■■■■■■■■■■■   ■■■■■■■■■■■■■■■■■■■■■■■■■■■■
```

Notification Flow

#### When App is Closed/Background

1. BackgroundAlarmService schedules local notification

2. System triggers notification at scheduled time

3. Notification appears in device notification tray

4. User taps notification

5. App opens to NotificationScreen

6. Alarm plays

#### When App is Open/Foreground

1. ForegroundAlarmMonitor checks time every second

2. Matches reminder time

3. Automatically opens NotificationScreen

4. Alarm plays

5. No system notification needed

Notification Configuration

Android (`AndroidManifest.xml`):

```
<uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
<uses-permission android:name="android.permission.SCHEDULE_EXACT_ALARM" />
<uses-permission android:name="android.permission.USE_FULL_SCREEN_INTENT" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.VIBRATE" />
```

iOS (`Info.plist`):

```
<key>UIBackgroundModes</key>
<array>
    <string>audio</string>
    <string>fetch</string>
    <string>remote-notification</string>
</array>
```

Notification Details

Channel Configuration:

• **Channel ID**: `pill_reminder_channel`

• **Importance**: Max (Android)

• **Priority**: High

• **Sound**: Enabled

• **Vibration**: Enabled

• **Full Screen Intent**: Enabled (pops up on lock screen)

iOS Configuration:

- **Presentation**: Alert, Badge, Sound
- **Interruption Level**: Time-sensitive
- **Sound**: Default system sound

Localization

Supported Languages

1. **English (en)** - Default
2. **Japanese (ja)** - Primary target market
3. **Chinese (zh)** - Additional support

Localization Files

- `lib/l10n/app_en.arb` - English strings (473 entries)
- `lib/l10n/app_ja.arb` - Japanese strings (338 entries)
- `lib/l10n/app_zh.arb` - Chinese strings (partial)

Key Translation Categories

- Authentication & Onboarding
- Medical Services (150+ service types)
- Locations (47 prefectures, cities, wards)
- Reminder System (40+ strings)
- Chatbot Interface
- UI Elements

Usage Example

```
final l10n = AppLocalizations.of(context)!;
Text(l10n.medicineReminderTitle);  // "Medicine Reminders" or "■■■■■■■■"
```

Dynamic Content

- Service names localized by type
- Prefecture/city names translated
- Medical categories in all languages
- Error messages localized

Dependencies

Core Dependencies

#### UI & Framework

```
flutter:
  sdk: flutter
flutter_localizations:
  sdk: flutter
cupertino_icons: ^1.0.8
```

#### Firebase & Authentication

```
firebase_core: ^4.1.1
firebase_auth: ^6.1.0
```

#### Location & Maps

```
google_maps_flutter: ^2.5.0
geolocator: ^14.0.2
latlong2: ^0.9.1
flutter_map: ^8.2.2
```

#### Notifications & Alarms

```
flutter_local_notifications: ^19.5.0
timezone: ^0.10.1
audioplayers: ^6.1.0
```

#### Storage & Data

```
shared_preferences: ^2.3.4
```

#### Media & Input

```
image_picker: ^1.2.1
speech_to_text: ^7.0.0
flutter_tts: ^4.2.0
permission_handler: ^11.3.1
```

#### Network & Communication

```
http: ^1.6.0
url_launcher: ^6.3.2
flutter_phone_direct_caller: ^2.1.1
```

#### Utilities

```
intl: any
flutter_dotenv: ^5.2.1
```

Dev Dependencies

```
flutter_test:
  sdk: flutter
flutter_lints: ^5.0.0
flutter_native_splash: ^2.4.6
```

Setup & Installation

Prerequisites

1. **Flutter SDK**: 3.8.0 or higher

2. **Dart SDK**: 3.8.0 or higher

3. **IDE**: VS Code or Android Studio

4. **Platform SDKs**:

• Android Studio with Android SDK

• Xcode (for iOS development on macOS)

Installation Steps

1. **Clone Repository**

```
git clone <repository-url>
cd medinavi-mac
```

2. **Install Dependencies**

```
flutter pub get
```

3. **Configure Environment**

Create `.env` file in project root:

```
GOOGLE_MAPS_API_KEY=your_api_key_here
GEMINI_API_KEY=your_gemini_api_key_here
```

4. **Generate Localization Files**

```
flutter gen-l10n
```

5. **Configure Firebase**

• Add `google-services.json` (Android) to `android/app/`

• Add `GoogleService-Info.plist` (iOS) to `ios/Runner/`

6. **Run the App**

```
flutter run
```

Platform-Specific Setup

#### Android

1. Update `android/app/build.gradle`:

```
minSdkVersion 21
targetSdkVersion 34
```

2. Enable permissions in `AndroidManifest.xml` (already configured)

#### iOS

1. Update `ios/Podfile`:

```
platform :ios, '12.0'
```

2. Install pods:

```
cd ios
pod install
```

3. Configure capabilities in Xcode:

- Background Modes (Audio, Remote Notifications)
- Push Notifications
- Location When In Use

Configuration

Google Maps API

1. Enable Google Maps SDK for Android/iOS
2. Enable Places API
3. Enable Directions API
4. Add API key to `.env` and platform-specific files

Firebase Configuration

1. Create Firebase project
2. Enable Authentication (Email/Password)
3. Download configuration files
4. Add to project

Notification Sounds

Place alarm sound file at:

```
assets/sounds/alarm.mp3
```

Update `pubspec.yaml`:

```
assets:
  - assets/sounds/
```

Usage Guide

For End Users

#### Setting Up Medicine Reminders

1. **Navigate to Reminders Tab**

• Tap "Reminders" in bottom navigation

2. **Add New Reminder**

• Tap the "+" button

• Enter medicine name

• Select time

• Choose repeat pattern

• (Optional) Set meal timing

• (Optional) Add medicine photo

• Tap "Add"

3. **Managing Reminders**

• Toggle switch to enable/disable

• Tap edit icon to modify

• Swipe left to delete

• Tap alarm icon to test

4. **When Alarm Triggers**

• Screen shows medicine details

• Tap "I took it" to mark as taken

• Or tap "Close" to dismiss

#### Finding Medical Services

1. **Select Location**

• Tap location dropdown at top

• Choose prefecture, city, ward

• Or tap "Use Current Location"

2. **Filter by Category**

• Swipe through category tabs

• Select desired service type

3. **View Results**

• See list of nearby facilities

• Distance shown for each

• Tap for more details

4. **Take Action**

• Tap phone icon to call

• Tap map icon for directions

• Tap heart to save to favorites

#### Using AI Chatbot

1. **Open Chatbot**

- Tap "Home" tab

- Scroll to chatbot section

2. **Describe Symptoms**

- Type or speak your symptoms

- Be specific about location and severity

3. **Get Recommendations**

- Review AI suggestions

- See recommended departments

- Tap "Find Nearby Facilities"

4. **Switch Response Mode**

- Toggle Quick/Detailed mode

- Quick: Brief answers

- Detailed: Comprehensive info

For Developers

#### Adding New Medical Services

Edit `lib/data/medical_services_data.dart`:

```
MedicalService(
  id: 'unique_id',
  name: 'Service Name',
  category: 'categoryGeneralServices',
  serviceType: 'serviceGeneralMedicine',
  latitude: 35.6762,
  longitude: 139.6503,
  address: 'Full address',
  phone: '+81-xx-xxxx-xxxx',
  website: 'https://example.com',
  isOpen: true,
)
```

#### Adding Localization Strings

1. Add to `lib/l10n/app_en.arb`:

```
"keyName": "English text"
```

2. Add to `lib/l10n/app_ja.arb`:

```
"keyName": "■■■■■■■"
```

3. Regenerate:

```
flutter gen-l10n
```

4. Use in code:

```
Text(l10n.keyName)
```

#### Customizing Notification Behavior

Edit `lib/services/notification_service.dart`:

```
// Change notification priority
importance: Importance.max,

// Change sound
sound: 'custom_sound.mp3',

// Change vibration pattern
enableVibration: true,
```

API Integration

Google Places API

Base URL: `https://maps.googleapis.com/maps/api/place/`

Endpoints Used:

1. **Nearby Search**

```
GET /nearbysearch/json
Parameters:
  - location: lat,lng
  - radius: meters
  - type: hospital|pharmacy|doctor
  - key: API_KEY
```

2. **Place Details**

```
GET /details/json
Parameters:
  - place_id: string
  - fields: name,rating,formatted_phone_number
  - key: API_KEY
```

Gemini AI API

Purpose: Power the medical chatbot

Base URL: `https://generativelanguage.googleapis.com/`

Request Format:

```
{
  "contents": [{
    "parts": [{
      "text": "User's medical question"
    }]
  }],
  "generationConfig": {
    "temperature": 0.7,
    "maxOutputTokens": 500
  }
}
```

Response Handling:

• Parse AI recommendations

• Extract medical department suggestions

• Format for user display

Testing

Manual Testing Checklist

#### Reminder System

- [ ] Create reminder with all repeat types
- [ ] Test notifications on lock screen
- [ ] Test notifications when app is closed
- [ ] Test alarm sound playback
- [ ] Verify "I took it" marks reminder as taken
- [ ] Test meal timing confirmations
- [ ] Verify reminders persist after app restart
- [ ] Test reminder edit and delete

#### Medical Services

- [ ] Search by current location
- [ ] Search by prefecture/city/ward
- [ ] Filter by different categories
- [ ] Verify distance calculations
- [ ] Test call functionality
- [ ] Test directions to Google Maps
- [ ] Add/remove favorites
- [ ] Verify favorites persist

#### Chatbot

- [ ] Test voice input
- [ ] Test text input
- [ ] Switch between Quick/Detailed modes
- [ ] Verify department recommendations
- [ ] Test facility search from chat
- [ ] Clear chat history

#### Localization

- [ ] Switch to Japanese
- [ ] Switch to English
- [ ] Verify all screens translate
- [ ] Check date/time formatting
- [ ] Verify language persists

Automated Testing

Create test files in `test/` directory:

```
// test/reminder_test.dart
void main() {
  group('Reminder Tests', () {
    test('Create reminder', () {
      final reminder = Reminder(
        id: '1',
        medicineName: 'Test Medicine',
        time: '08:00',
      );
      expect(reminder.medicineName, 'Test Medicine');
    });
  });
}
```

Run tests:

```
flutter test
```

Troubleshooting

Common Issues

#### Notifications Not Showing

Symptoms: Reminders set but no notifications appear

Solutions:

1. Check device notification permissions

• Android: Settings → Apps → MediNavi → Notifications

• iOS: Settings → MediNavi → Notifications

2. Disable Do Not Disturb mode

3. Verify alarm sound file exists

4. Check app has background execution permission

5. Rebuild app:

```bash

flutter clean

flutter pub get

flutter run

```

#### Location Services Not Working

Symptoms: "Use Current Location" doesn't work

Solutions:

1. Enable location services

2. Grant location permission to app

3. Check GPS is enabled

4. Verify Google Maps API key is valid

5. Test on real device (emulators may have issues)

#### Google Maps Not Displaying

Symptoms: Map screen shows blank or error

Solutions:

1. Verify API key in `.env` file

2. Enable Maps SDK in Google Cloud Console

3. Check API key restrictions

4. Ensure billing is enabled on Google Cloud

5. Wait 5-10 minutes for API key to propagate

#### Chatbot Not Responding

Symptoms: AI assistant doesn't reply

Solutions:

1. Check internet connection

2. Verify Gemini API key

3. Check API quota limits

4. Review API endpoint URL

5. Check response timeout settings

Future Enhancements

Planned Features

#### Short-term (1-3 months)

1. **Reminder Improvements**

- Dose quantity tracking
- Refill reminders
- Medicine interaction warnings
- Export reminder history

2. **UI/UX Enhancements**

- Dark mode support
- Customizable themes
- Widget for home screen
- Quick actions

3. **Social Features**

- Share favorite facilities
- Review and rate facilities
- Emergency contacts

#### Medium-term (3-6 months)

1. **Health Tracking**

- Symptom diary
- Medication logs
- Appointment scheduler
- Health metrics tracking

2. **Advanced Notifications**

- Smart snooze (dynamic rescheduling)
- Weather-based medicine reminders
- Interaction with health apps

3. **Cloud Sync**

- Multi-device sync
- Cloud backup
- Family sharing

#### Long-term (6+ months)

1. **AI Enhancements**

- Personalized health insights
- Predictive reminders
- Health trend analysis
- Telemedicine integration

2. **Wearable Integration**

- Apple Watch app
- Android Wear support
- Fitbit integration

3. **Healthcare Provider Features**

- Doctor portal
- Prescription management
- Direct appointment booking
- Medical records integration

Technical Improvements

1. **Architecture**

- Migrate to BLoC pattern

- Implement clean architecture
- Add dependency injection

2. **Performance**

- Optimize map rendering
- Implement caching strategies
- Reduce app size

3. **Testing**

- Unit test coverage >80%
- Integration tests
- UI/E2E tests
- Performance benchmarks

4. **DevOps**

- CI/CD pipeline
- Automated releases
- Crash reporting
- Analytics integration

Security & Privacy

Data Storage

- **Local**: SharedPreferences (encrypted on device)
- **Cloud**: Firebase Authentication (secure)
- **Sensitive Data**: Medicine photos stored locally only

Permissions

- **Location**: Required for finding nearby facilities
- **Camera**: Optional, for medicine photos
- **Microphone**: Optional, for voice chatbot
- **Notifications**: Required for reminders
- **Phone**: Optional, for calling facilities

Privacy Policy

- No personal health data sent to servers
- Medicine information stored locally only
- Location data not stored persistently

- Firebase handles authentication securely
- Google Maps data subject to Google's privacy policy

Contributing

Development Workflow

1. Fork the repository
2. Create feature branch (`git checkout -b feature/AmazingFeature`)
3. Commit changes (`git commit -m 'Add AmazingFeature'`)
4. Push to branch (`git push origin feature/AmazingFeature`)
5. Open Pull Request

Code Style

- Follow Dart style guide
- Use `flutter analyze` before committing
- Format code with `dart format`
- Add comments for complex logic
- Write descriptive commit messages

Pull Request Guidelines

- Describe changes clearly
- Include screenshots for UI changes
- Update documentation
- Add tests if applicable
- Ensure all tests pass

License

This project is licensed under a proprietary license. All rights reserved.

Contact & Support

Developer: S1ThuHtun

Email: sithuhtunn97@gmail.com

GitHub: [Repository Link]

Reporting Issues

1. Check existing issues first
2. Provide detailed description
3. Include reproduction steps
4. Add screenshots/logs if applicable
5. Specify device and OS version

Acknowledgments

Technologies Used

- **Flutter Team**: For the amazing framework
- **Firebase**: For authentication services
- **Google**: For Maps and Places APIs
- **OpenStreetMap**: For map tiles
- **Gemini AI**: For chatbot capabilities

Third-Party Libraries

Special thanks to the maintainers of:

- flutter_local_notifications
- geolocator
- shared_preferences
- audioplayers
- And all other dependencies

Changelog

Version 1.0.0 (January 2026)

Initial Release

- Medicine reminder system with notifications
- Medical facility finder with Google Maps

- AI-powered medical chatbot
- Multi-language support (EN, JA, ZH)
- Favorites system
- User authentication

Recent Updates

- Added localization for reminder test alarm tooltip
- Fixed notification permissions for Android 13+
- Improved alarm reliability on iOS
- Enhanced meal timing confirmation dialogs
- Optimized location search performance

Appendix

A. Medical Service Categories

1. **General Services** (6 types)

- General Medicine, Emergency Care, Family Medicine
- Preventative Medicine, Health Check-ups, Urgent Care

2. **Specialized Services** (14 types)

- Cardiology, Dermatology, Endocrinology
- Gastroenterology, Hematology, Nephrology
- Neurology, OB/GYN, Ophthalmology
- Orthopedics, Pediatrics, Psychiatry
- Rheumatology, Pulmonology, Urology

3. **Surgical Services** (8 types)

- General Surgery, Cardiac Surgery
- Orthopedic Surgery, Neurosurgery
- Plastic Surgery, Pediatric Surgery
- Obstetric Surgery, Trauma Surgery

4. **Rehabilitation Services** (5 types)

- Physical Therapy, Occupational Therapy
- Speech Therapy, Cardiac Rehabilitation
- Neurological Rehabilitation

5. **Diagnostic Services** (6 types)

- Laboratory Tests, Radiology, Pathology

• Endoscopy, ECG, EEG

6. **Emergency Services** (4 types)

• Emergency Room, Trauma Care, ICU, Burn Unit

7. **Maternity & Women's Health** (4 types)

• Obstetrics, Gynecology

• Family Planning, Breast Health

8. **Pharmacy Services** (5 types)

• Prescription Medications, OTC Medications

• Vaccinations, Consultations, Compounding

9. **Mental Health Services** (5 types)

• Counseling, Psychotherapy

• Addiction Treatment, Support Groups

• Behavioral Therapy

10. **Alternative Medicine** (5 types)

• Acupuncture, Chiropractic Care

• Massage Therapy, Naturopathy, Homeopathy

B. Supported Locations

47 Prefectures including:

• Tokyo, Osaka, Kyoto, Hokkaido, Fukuoka

• Kanagawa, Saitama, Chiba, Aichi, Hyogo

• And 37 more prefectures

Major Cities including:

• Shibuya, Shinjuku, Chiyoda (Tokyo)

• Umeda, Namba, Tennoji (Osaka)

• Gion, Kyoto Station (Kyoto)

• And many more

C. Notification Settings Reference

Android Notification Details:

```
Importance: IMPORTANCE_MAX
Priority: PRIORITY_MAX
Category: CATEGORY_ALARM
Sound: Default or Custom
Vibration: Enabled
LED: Enabled (if device supports)
Lock Screen: Full screen intent
Channel: pill_reminder_channel
```

iOS Notification Details:

```
Alert: Enabled
Sound: Default
Badge: Enabled
```

```
    Interruption Level: TimeSensitive
    Authorization: Alert, Sound, Badge
    Critical Alerts: Not used
```

Document Version: 1.0

Last Updated: January 14, 2026

Status: Production Ready

This documentation is maintained alongside the codebase. For the latest updates, please refer to the repository.