

Firestore Cloud Messaging (FCM) Kurulum Rehberi

Bu doküman, Yardım Yolda uygulamasında Firestore Cloud Messaging (FCM) push notification'larının nasıl kurulacağını ve yapılandırılacağını açıklar.

İçindekiler

- Genel Bakış
- Placeholder Yapılandırma
- Gerçek Firestore Projesi Kurulumu
- FCM Özellikleri
- Teknik Detaylar
- Hata Ayıklama
- Test Etme
- Üretim Kontrol Listesi







Genel Bakış

Yardım Yolda uygulaması, kullanıcılara servis taleplerinin durum değişiklikleri hakkında bildirim göndermek için Firestore Cloud Messaging (FCM) kullanır.

Bildirim Akışı

```
Servis Talebi Durum Değişikliği (Supabase)
↓
Supabase Realtime Event
↓
ServiceNotificationListener (Flutter)
↓
NotificationService (FCM + Local Notifications)
↓
Kullanıcının Cihazında Bildirim
```

Ana Özellikler

-  Otomatik bildirim izni isteği (OTP girişi sonrası)
-  FCM token yönetimi ve veritabanına kaydetme
-  Supabase Realtime ile gerçek zamanlı durum dinleme
-  Foreground, background ve terminated durumlarında bildirim gösterme
-  Türkçe bildirim mesajları
-  Graceful error handling

Placeholder Yapılandırma

Proje, geliştirme kolaylığı için placeholder (sahte) Firestore yapılandırma dosyalarıyla gelir:

Android

```
android/app/google-services.json
```

iOS

```
ios/Runner/GoogleService-Info.plist
```

⚠ **Önemli:** Bu dosyalar sadece projenin derlenmesini sağlar. Gerçek bildirim göndermek için gerçek Firebase projesi yapılandırması gereklidir.

Placeholder ile Çalışma

Placeholder yapılandırma ile:

- ✓ Proje derlenebilir
- ✓ Uygulama çalışır
- ✓ Supabase Realtime bildirimleri çalışır (local notifications)
- ✗ FCM push notifications çalışmaz
- ✗ Remote notifications çalışmaz
- ✗ Background/terminated state notifications çalışmaz

Placeholder Dosyalarının Yapısı

google-services.json:

```
{
  "_comment": "⚠ PLACEHOLDER CONFIGURATION - NOT FOR PRODUCTION USE ⚠",
  "project_info": {
    "project_id": "yardimyolda-placeholder",
    ...
  },
  "client": [...]
}
```

GoogleService-Info.plist:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ⚠ PLACEHOLDER CONFIGURATION - NOT FOR PRODUCTION USE ⚠ -->
<dict>
  <key>PROJECT_ID</key>
  <string>yardimyolda-placeholder</string>
  ...
</dict>
```



Gerçek Firebase Projesi Kurulumu

Adım 1: Firebase Projesi Oluşturma

1. [Firebase Console](https://console.firebase.google.com/) (https://console.firebase.google.com/) adresine gidin
2. “Add project” butonuna tıklayın
3. Proje adını girin: `yardimyolda-prod` (veya tercih ettiğiniz isim)
4. Google Analytics’i etkinleştirin (isteğe bağlı ama önerilir)

5. **“Create project”** butonuna tıklayın
6. Proje oluşturulmasını bekleyin (1-2 dakika)

Adım 2: Android Uygulaması Ekleme

2.1. Firebase Console’da Android Uygulaması Ekle

1. Firebase Console’da projenize gidin
2. Project Overview sayfasında **Android** simgesine tıklayın
3. Android package name girin: `com.yardimyolda.app`
4. App nickname girin (isteğe bağlı): `Yardım Yolda Android`
5. Debug signing certificate SHA-1 ekleyin (isteğe bağlı, geliştirme için önerilir)
6. **“Register app”** butonuna tıklayın

2.2. SHA-1 Fingerprint Alma

Debug Keystore için (Geliştirme):

```
keytool -list -v \
  -keystore ~/.android/debug.keystore \
  -alias androiddebugkey \
  -storepass android \
  -keypass android
```

Production Keystore için (Üretim):

```
keytool -list -v \
  -keystore /path/to/your/release-keystore.jks \
  -alias your-key-alias
```

SHA-1 fingerprint’i kopyalayın ve Firebase Console’da “Add fingerprint” kısmına yapıştırın.

2.3. google-services.json İndirme

1. **“Download google-services.json”** butonuna tıklayın
2. İndirilen dosyayı projenizde şu konuma kopyalayın: `android/app/google-services.json`
3. Mevcut placeholder dosyasının üzerine yazın

2.4. Yapılandırmayı Doğrulama

Dosya içeriğini kontrol edin:

```
cat android/app/google-services.json
```

project_id’nin gerçek olduğundan emin olun:

```
{
  "project_info": {
    "project_id": "yardimyolda-prod", // ← "placeholder" olmamalı
    ...
  }
}
```

Adım 3: iOS Uygulaması Ekleme

3.1. Firebase Console'da iOS Uygulaması Ekle

1. Firebase Console'da projenize gidin
2. Project Overview sayfasında **iOS** simgesine tıklayın
3. iOS bundle ID girin: `com.yardimyolda.app`
4. App nickname girin (isteğe bağlı): `Yardım Yol'da iOS`
5. **"Register app"** butonuna tıklayın

3.2. GoogleService-Info.plist İndirme

1. **"Download GoogleService-Info.plist"** butonuna tıklayın
2. İndirilen dosyayı projenizde şu konuma kopyalayın:
`ios/Runner/GoogleService-Info.plist`
3. Mevcut placeholder dosyasının üzerine yazın

3.3. Xcode'da Projeye Ekleme

1. Xcode'u açın
2. `ios/Runner.xcworkspace` dosyasını açın
3. Project Navigator'da `Runner` klasörünü seçin
4. `GoogleService-Info.plist` dosyasını sürükleyip bırakın
5. **"Copy items if needed"** seçeneğini işaretleyin
6. **Target:** `Runner` seçili olduğundan emin olun
7. **"Finish"** butonuna tıklayın

3.4. APNs Sertifikası Yapılandırma (Üretim için gerekli)

Development APNs Sertifikası:

1. Apple Developer Portal'a gidin
2. Certificates, Identifiers & Profiles > Certificates
3. **"+"** butonuna tıklayın
4. **"Apple Push Notification service SSL (Sandbox & Production)"** seçin
5. Bundle ID olarak `com.yardimyolda.app` seçin
6. CSR dosyası oluşturun ve yükleyin
7. Sertifikayı indirin ve Keychain'e ekleyin

Firebase'e APNs Key Ekleme:

1. Firebase Console > Project Settings > Cloud Messaging
2. **"APNs Authentication Key"** bölümüne gidin
3. **"Upload"** butonuna tıklayın
4. Apple Developer Portal'dan aldığınız .p8 key dosyasını yükleyin
5. Key ID ve Team ID'yi girin

Adım 4: Yapılandırmayı Doğrulama

4.1. Dependencies'leri Yükle

```
cd /path/to/yardimyolda_flutter
flutter clean
flutter pub get
```

4.2. Android Build Test

```
# Debug build
flutter build apk --debug

# Release build
flutter build apk --release
```

Beklenen çıktı:

```
✓ Built build/app/outputs/flutter-apk/app-debug.apk
```

4.3. iOS Build Test (macOS)

```
# Debug build
flutter build ios --debug --no-codesign

# Release build
flutter build ios --release --no-codesign
```

Beklenen çıktı:

```
✓ Built build/ios/iphoneos/Runner.app
```

4.4. Çalışma Zamanı Doğrulama

Uygulamayı çalıştırın ve debug console'u kontrol edin:

```
flutter run
```

Beklenen log çıktıları:

```
✓ Firebase başarıyla başlatıldı
✓ Notification service başarıyla başlatıldı
🔔 Bildirim izinleri isteniyor...
✓ Bildirim izni verildi
✓ FCM token alındı: eyJhbGciOiJSUzI1NiIsImtpZCI6...
✓ FCM token veritabanına kaydedildi
🎧 Servis bildirimleri dinleniyor: user-uuid-here
✓ ServiceNotificationListener başarıyla başlatıldı
```

Placeholder config ile çalışıyorsa:

```
⚠ Firebase başlatma hatası (placeholder config kullanılıyor)
⚠ Üretim için gerçek Firebase yapılandırması gereklidir
```



FCM Özellikleri

1. Otomatik Bildirim İzni

Kullanıcı OTP ile giriş yaptığında otomatik olarak bildirim izni istenir.

Kod konumu: lib/features/auth/providers/auth_state_provider.dart

```
Future<void> verifyOTP(...) async {
  // ... OTP verification ...

  // Request notification permissions
  _requestNotificationPermissionsAndSaveToken(user.id);

  // Start service notification listener
  _startServiceNotificationListener(user.id);
}
```

2. FCM Token Yönetimi

Token otomatik olarak alınır ve Supabase'e kaydedilir.

Veritabanı şeması:

```
-- user_profiles tablosuna device_token eklendi
ALTER TABLE user_profiles
ADD COLUMN device_token TEXT;
```

Token kaydetme:

```
await _repository.updateDeviceToken(
  userId: userId,
  deviceToken: token,
);
```

3. Supabase Realtime Entegrasyonu

service_requests tablosundaki status değişiklikleri dinlenir.

Kod konumu: lib/core/services/service_notification_listener.dart

```
_channel!.onPostgresChanges(
  event: PostgresChangeEvent.update,
  schema: 'public',
  table: 'service_requests',
  filter: PostgresChangeFilter(
    type: PostgresChangeFilterType.eq,
    column: 'customer_id',
    value: userId,
  ),
  callback: (payload) => _handleServiceRequestUpdate(payload),
)
```

4. Bildirim Mesajları

Durum değişikliklerine göre Türkçe mesajlar gösterilir:

Status	Türkçe Mesaj
pending	"Çekici talebiniz alındı. Sağlayıcı aranıyor..."
accepted	"Çekici talebiniz kabul edildi!"
on_the_way	"Sağlayıcı yola çıktı. Yakında yanınızda olacak."
in_progress	"Çekici hizmeti başladı."
completed	"Çekici hizmeti tamamlandı. Lütfen değerlendirin."
cancelled	"Çekici talebiniz iptal edildi."

Mesaj oluşturma:

```
String _getStatusMessage(String serviceType, String status) {
  switch (status.toLowerCase()) {
    case 'accepted':
      return '$serviceType talebiniz kabul edildi!';
    // ...
  }
}
```

Teknik Detaylar

Kullanılan Paketler

```
dependencies:
  firebase_core: ^2.27.0          # Firebase SDK
  firebase_messaging: ^14.7.19    # FCM
  flutter_local_notifications: ^17.0.0 # Local notifications
```

Servis Mimarisi

```
NotificationService (Singleton)
├─ Firebase initialization
├─ FCM token management
├─ Permission handling
└─ Local notification display

ServiceNotificationListener (Singleton)
├─ Supabase Realtime channel
├─ PostgreSQL change detection
└─ Notification triggering

NotificationProvider (Riverpod)
├─ State management
├─ Permission state
└─ Token state
```

Dosya Yapısı

```
lib/
├── core/
│   ├── services/
│   │   ├── notification_service.dart          # FCM & Local Notifications
│   │   └── service_notification_listener.dart # Supabase Realtime
│   └── providers/
│       ├── notification_provider.dart          # Riverpod providers
│       └── service_notification_provider.dart # Listener state
├── features/
│   └── auth/
│       └── providers/
│           └── auth_state_provider.dart        # Auth + Notification integration
└── main.dart                                  # Firebase initialization
```

Android Yapılandırma

AndroidManifest.xml:

```
<!-- FCM Permissions -->
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.POST_NOTIFICATIONS" />

<!-- FCM Metadata -->
<meta-data
    android:name="com.google.firebase.messaging.default_notification_channel_id"
    android:value="yardimyolda_channel" />
```

build.gradle:

```
dependencies {
    implementation platform('com.google.firebase:firebase-bom:32.7.0')
    implementation 'com.google.firebase:firebase-messaging'
}
```

iOS Yapılandırma

Info.plist:

```
<!-- Notification Permissions -->
<key>NSUserNotificationAlertStyle</key>
<string>alert</string>

<!-- Background Modes -->
<key>UIBackgroundModes</key>
<array>
    <string>remote-notification</string>
</array>
```

AppDelegate.swift:


```

override func application(
    _ application: UIApplication,
    didFinishLaunchingWithOptions launchOptions: ...
) -> Bool {
    application.registerForRemoteNotifications()
    return super.application(application, didFinishLaunchingWithOptions: launchOptions)
}

```



Hata Ayıklama

Firebase Başlatma Hatası

Hata:

⚠️ Firebase başlatma hatası (placeholder config kullanılıyor)

Çözüm:

1. `google-services.json` ve `GoogleService-Info.plist` dosyalarının gerçek config olduğundan emin olun
2. Dosyalardaki `project_id` alanını kontrol edin
3. `flutter clean` & `flutter pub get` çalıştırın
4. Uygulamayı yeniden build edin

Token Alınamıyor

Android Debug:

```

# FCM loglarını filtrele
adb logcat | grep -i fcm

# Firebase loglarını filtrele
adb logcat | grep -i firebase

```

iOS Debug:

1. Xcode'u açın
2. Window > Devices and Simulators
3. Cihazı seçin
4. "View Device Logs"
5. "firebase" veya "fcm" ile filtreleyin

Olası nedenler:

- Google Play Services yüklü değil (Android)
- APNs sertifikası yüklenmemiş (iOS)
- Internet bağlantısı yok
- Firebase config hatalı

Bildirimler Görünmüyor

Kontrol listesi:

1. ☒ Bildirim izni verildi mi?


dart

```
final settings = await FirebaseMessaging.instance.getNotificationSettings();
print('Authorization: ${settings.authorizationStatus}');
```

1.  FCM token alındı mı?


```
dart
```

```
final token = await NotificationService().getDeviceToken();
print('Token: $token');
```

2.  Token veritabanına kaydedildi mi?

```
sql
```

```
SELECT device_token FROM user_profiles WHERE id = 'user-uuid';
```

3.  Supabase Realtime bağlantısı aktif mi?

```
dart
```

```
print('Listening: ${_serviceNotificationListener.isListening}');
```

4.  Android notification channel oluşturuldu mu?

```
dart
```

```
final channels = await _localNotifications
    .resolvePlatformSpecificImplementation<AndroidFlutterLocalNotificationsPlugin>()
    ?.getNotificationChannels();
print('Channels: $channels');
```

Supabase Realtime Çalışmıyor

Debug:

```
supabase
  .channel('test-channel')
  .onPostgresChanges(
    event: PostgresChangeEvent.all,
    schema: 'public',
    table: 'service_requests',
    callback: (payload) {
      print('📡 Realtime event: $payload');
    },
  )
  .subscribe((status, error) {
    print('📡 Status: $status');
    if (error != null) print('❌ Error: $error');
  });
```

Olası nedenler:

- Supabase Realtime aktif değil
- RLS politikaları engelleme yapıyor
- Network bağlantısı kesik

✓ Test Etme

1. Manuel Token Test

```
// lib/features/auth/presentation/pages/test_fcm_page.dart

import 'package:flutter/material.dart';
import '../../../../../core/services/notification_service.dart';

class TestFCMPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('FCM Test')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            ElevatedButton(
              onPressed: () async {
                final token = await NotificationService().getDeviceToken();
                print('🔑 FCM Token: $token');
                ScaffoldMessenger.of(context).showSnackBar(
                  SnackBar(content: Text('Token: ${token?.substring(0, 20)}...')),
                );
              },
              child: Text('Get FCM Token'),
            ),
            SizedBox(height: 16),
            ElevatedButton(
              onPressed: () async {
                await NotificationService().showNotification(
                  title: 'Test Bildirimi',
                  body: 'Bu bir test bildirimidir',
                );
              },
              child: Text('Show Test Notification'),
            ),
          ],
        ),
      ),
    );
  }
}
```

2. Supabase Realtime Test

SQL Editor'de manuel status güncelleme:

```
-- Test için bir service request oluştur
INSERT INTO service_requests (
  customer_id,
  service_type,
  status,
  customer_lat,
  customer_lng
) VALUES (
  'your-user-id',
  'towing',
  'pending',
  41.0082,
  28.9784
);

-- Status'u güncelle (bildirim tetiklenir)
UPDATE service_requests
SET status = 'accepted'
WHERE customer_id = 'your-user-id'
AND status = 'pending';

-- Farklı durumlar dene
UPDATE service_requests
SET status = 'on_the_way'
WHERE customer_id = 'your-user-id';
```

3. Otomatik Test

```
// test/services/notification_service_test.dart

import 'package:flutter_test/flutter_test.dart';
import 'package:yardimyolda/core/services/notification_service.dart';

void main() {
  group('NotificationService', () {
    test('should initialize successfully', () async {
      final service = NotificationService();
      await service.initialize();
      expect(service, isNotNull);
    });

    test('should generate correct status message', () {
      final service = NotificationService();
      final message = service._getStatusMessage('Çekici', 'accepted');
      expect(message, contains('kabul edildi'));
    });
  });
}
```

Üretim Kontrol Listesi

Firestore Yapılandırma

- [] Gerçek Firestore projesi oluşturuldu
- [] google-services.json gerçek config ile değiştirildi
- [] GoogleService-Info.plist gerçek config ile değiştirildi
- [] Firestore Console'da Android uygulaması eklendi

- [] Firebase Console'da iOS uygulaması eklendi

Android

- [] Production keystore SHA-1 fingerprint Firebase'e eklendi
- [] `AndroidManifest.xml` FCM permissions içeriyor
- [] `build.gradle` Firebase dependencies içeriyor
- [] Google Play Services güncel

iOS

- [] APNs sertifikası oluşturuldu
- [] APNs key Firebase'e yüklendi
- [] `Info.plist` notification permissions içeriyor
- [] `AppDelegate.swift` yapılandırıldı
- [] Xcode'da `GoogleService-Info.plist` eklendi

Uygulama

- [] Bildirim izinleri test edildi
- [] FCM token alınıyor ve veritabanına kaydediliyor
- [] Supabase Realtime bağlantısı çalışıyor
- [] Foreground notifications görünüyor
- [] Background notifications görünüyor
- [] Terminated state notifications görünüyor
- [] Notification tap handling çalışıyor
- [] Türkçe mesajlar doğru görünüyor

Test Senaryoları

- [] Yeni kullanıcı kaydı → Bildirim izni isteniyor
- [] Servis talebi oluşturma → Status: pending → Bildirim gösteriliyor
- [] Status güncelleme: accepted → Bildirim gösteriliyor
- [] Status güncelleme: on_the_way → Bildirim gösteriliyor
- [] Status güncelleme: completed → Bildirim gösteriliyor
- [] Logout → Service listener duruyor
- [] Login → Service listener yeniden başlıyor



İleri Seviye: Server-Side Bildirim Gönderimi

Şu anda bildirimler client-side (Supabase Realtime) ile çalışıyor. Server-side bildirim göndermek için:

Yaklaşım 1: Supabase Edge Function + Firebase Admin SDK

1. Edge Function Oluştur:

```
// supabase/functions/send-notification/index.ts

import { serve } from 'https://deno.land/std@0.168.0/http/server.ts'
import { createClient } from 'https://esm.sh/@supabase/supabase-js@2'

serve(async (req) => {
  const { userId, title, body } = await req.json()

  // Get user's FCM token from database
  const supabase = createClient(...)
  const { data: user } = await supabase
    .from('user_profiles')
    .select('device_token')
    .eq('id', userId)
    .single()

  // Send FCM message
  const response = await fetch('https://fcm.googleapis.com/fcm/send', {
    method: 'POST',
    headers: {
      'Authorization': `key=${Deno.env.get('FCM_SERVER_KEY')}`,
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({
      to: user.device_token,
      notification: { title, body },
    }),
  })

  return new Response(JSON.stringify({ success: true }), {
    headers: { 'Content-Type': 'application/json' },
  })
})
```

2. Database Trigger Oluştur:

```

CREATE OR REPLACE FUNCTION notify_service_request_update()
RETURNS TRIGGER AS $$
BEGIN
    -- Call Edge Function when status changes
    PERFORM
        net.http_post(
            url := 'https://your-project.supabase.co/functions/v1/send-notification',
            headers := jsonb_build_object(
                'Authorization', 'Bearer ' || current_setting('request.jwt.claim.sub')
            ),
            body := jsonb_build_object(
                'userId', NEW.customer_id,
                'title', 'Servis Talebi Güncellendi',
                'body', 'Talebinizin durumu değişti: ' || NEW.status
            )
        );
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER on_service_request_status_change
AFTER UPDATE OF status ON service_requests
FOR EACH ROW
WHEN (OLD.status IS DISTINCT FROM NEW.status)
EXECUTE FUNCTION notify_service_request_update();

```

Yaklaşım 2: Firebase Cloud Functions

```

// functions/src/index.ts

import * as functions from 'firebase-functions';
import * as admin from 'firebase-admin';

admin.initializeApp();

export const sendServiceNotification = functions.firestore
    .document('service_requests/{requestId}')
    .onUpdate(async (change, context) => {
        const before = change.before.data();
        const after = change.after.data();

        if (before.status === after.status) return;

        const userId = after.customer_id;
        const token = await getTokenFromSupabase(userId);

        const message = {
            notification: {
                title: 'Servis Talebi Güncellendi',
                body: getStatusMessage(after.service_type, after.status),
            },
            token: token,
        };

        await admin.messaging().send(message);
    });

```



Kaynaklar

- [Firebase Cloud Messaging](https://firebase.google.com/docs/cloud-messaging) (<https://firebase.google.com/docs/cloud-messaging>)
- [Flutter Firebase Messaging Plugin](https://pub.dev/packages/firebase_messaging) (https://pub.dev/packages/firebase_messaging)
- [Flutter Local Notifications](https://pub.dev/packages/flutter_local_notifications) (https://pub.dev/packages/flutter_local_notifications)
- [Supabase Realtime](https://supabase.com/docs/guides/realtime) (<https://supabase.com/docs/guides/realtime>)
- [Firebase Admin SDK](https://firebase.google.com/docs/admin/setup) (<https://firebase.google.com/docs/admin/setup>)



Destek

Sorun yaşıyorsanız:

1. Bu dokümandaki [Hata Ayıklama](#) bölümünü kontrol edin
2. Debug loglarını paylaşın
3. GitHub issues'da yeni bir issue açın

Son Güncelleme: 2025-10-23

Versiyon: 1.0.0