

**ANKARA ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ**

**BLM4537 – IOS ile Mobil Uygulama Geliştirme Dersi**

**Proje Raporu**



**PROJE KONUSU:**

**Şifreli Mesajlaşma Uygulaması**

**Ziya Eren Altaş**

**21290571**

**Github: <https://github.com/erenaltas72/ios>**

**Proje Videosu:**

**[https://drive.google.com/drive/folders/1s\\_lpcPcptWTb0K\\_Mxef8w4\\_8oLUoZEoW?usp=sharing](https://drive.google.com/drive/folders/1s_lpcPcptWTb0K_Mxef8w4_8oLUoZEoW?usp=sharing)**

# 1. Giriş

## 1.1 Proje Tanımı

SecureChat, şifreli mesajlaşma, dosya paylaşımı ve grup sohbetini destekleyen modern bir web tabanlı chat uygulamasıdır. Uygulama, Flutter ve .NET Core teknolojileri kullanılarak geliştirilmiş olup, kullanıcıların güvenli ve anlık iletişim kurmasını sağlamaktadır. Uygulama bilinen mesajlaşma uygulamalarının (örnek: whatsapp) özelliklerini barındırmakla birlikte popüler uygulamalarda olmasını istediğim kullanıcı özelliklerinin de eklenmiş bir halidir.

## 1.2 Proje Amacı

Bu proje, aşağıdaki hedefler doğrultusunda geliştirilmiştir:

- **Güvenli İletişim Sağlamak:** Tüm mesajlar base64 şifrelemesi ile korunmakta, kullanıcı gizliliği öncelenmiştir.
- **Modern UI/UX Deneyimi:** Flutter framework'ü kullanılarak responsive ve kullanıcı dostu arayüz oluşturmak.
- **Ölçeklenebilir Backend Mimarisi:** .NET Core REST API ile profesyonel ve bakım yapılabilir bir backend infrastrüktürü kurmak.
- **Gerçek Zamanlı Veri Yönetimi:** Veritabanı sorgularını optimize ederek performanslı bir sistem sağlamak.
- **Çoklu Dil Desteği:** Uygulama kullanıcılarına Türkçe ve İngilizce tercih etme seçeneği sunmak.
- **Kullanışlı Özellikler:** Popüler mesajlaşma uygulamalarında olmayan kullanışlı özellikler.

## 1.3 Proje Kapsamı

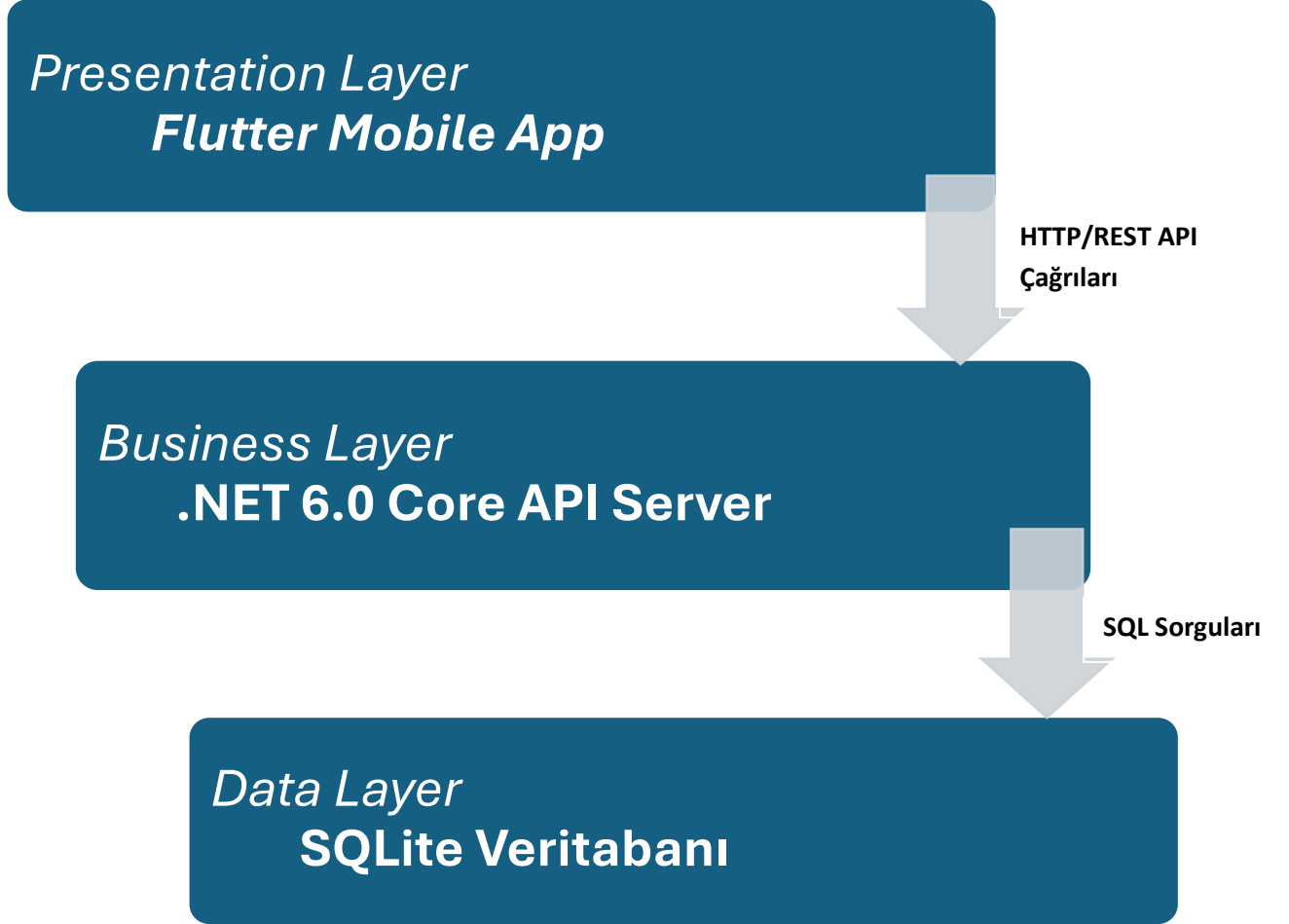
SecureChat aşağıdaki işlevselliği içermektedir:

- **Kullanıcı kaydı ve kimlik doğrulaması**
- **Kişisel ve grup sohbetleri**
- **Medya (Fotoğraf) ve dosya paylaşımı**
- **Mesajları düzenleme ve silme (herkesten ve kendinden)**
- **Mesaj yanıtlama**
- **Emoji reaksiyonları**
- **Mesajları yıldızlama ve sabitleme**
- **İngilizce dil desteği**
- **Koyu Tema**
- **Durum Paylaşımı**
- **Mesaj Sabitleme**
- **Profil düzenleme**
- **Kategorilere ayrılabilen yıldızlı mesajlar**

## 2. Sistem Mimarisi

### 2.1. Mimarinin Genel Yapısı

SecureChat uygulaması 3-Katmanlı (3-Tier) Mimarisi üzerine kurulmuştur:



### 2.2. Katmanların Detaylı Açıklaması

#### 2.2.1. Presentation Layer

Ana Bileşenler:

- main.dart - Uygulama giriş noktası, MaterialApp konfigürasyonu
- screens/chat\_page.dart - Ana sohbet ekranı, mesaj görüntüleme ve gönderme
- screens/group\_pages.dart - Grup oluşturma ve yönetimi
- screens/profile\_page.dart - Kullanıcı profili ve ayarlar
- screens/login\_page.dart - Giriş ve kayıt ekranı
- screens/media\_picker\_widget.dart - Medya seçimi (galeri, dosya)
- utils.dart - Şifreleme/deşifreleme fonksiyonları, yardımcı metodları

### **Teknolojiler:**

- Flutter 3.3.0+ (UI framework)
- Dart (programlama dili)
- http paketi (REST API çağrıları)
- image\_picker, file\_picker (medya seçimi)
- shared\_preferences (local veri depolaması)
- intl, flutter\_localizations (çokdil desteği)

### **Sorumlulukları:**

- Kullanıcı arayüzü (UI) oluşturma ve yönetme
- Kullanıcı girişini alma ve validasyon
- Backend API'ye HTTP istekleri gönderme
- JSON response'larını işleme ve UI güncelleme
- Medya (resim, dosya) seçme ve encoding
- Yerel veri depolaması (SharedPreferences)
- Dil ve yerelleştirme yönetimi

## **2.2.2.Business Layer**

### **Ana Bileşenler:**

- Program.cs - Uygulama başlatma, dependency injection konfigürasyonu
- Controllers/ - API endpoint'leri:
  - UserController.cs - Kullanıcı yönetimi (kayıt, giriş, profil)
  - MessagesController.cs - Mesaj gönderme, alma, silme, düzenleme
  - GroupsController.cs - Grup oluşturma, üye yönetimi
  - SettingsController.cs - Ayar yönetimi
  - StatusController.cs - Çevrimiçi durumu
  - NotificationController.cs - Bildirimler
- Models/ - Veri modelleri:
  - User.cs, Message.cs, Group.cs, GroupMember.cs
  - MessageReaction.cs, Status.cs, Notification.cs

- ChatContext.cs - Entity Framework DbContext
- Migrations/ - Veritabanı şema güncellemeleri

#### **Teknolojiler:**

- .NET 6.0+ (framework)
- C# 10+ (programlama dili)
- Entity Framework Core 6.0+ (ORM)
- ASP.NET Core (web framework)
- Linq (veri sorgulama)

#### **Sorumlulukları:**

- REST API endpoint'lerini sunma
- Veritabanı işlemleri (CRUD)
- İş mantığı uygulaması
- Veri validasyonu
- Authentication & Authorization
- Veritabanı migration yönetimi
- Error handling ve logging

### **2.2.3. Data Layer**

#### **Ana Tablolar:**

- Users - Kullanıcı hesapları
- Messages - Gönderilen/alınan mesajlar
- Groups - Grup sohbetleri
- GroupMembers - Grup üyeliği ilişkileri
- MessageReactions - Emoji reaksiyonlar
- Status - Kullanıcı çevrimiçi/çevrimdışı durumu
- Notifications - Bildirim geçmişi

#### **Sorumlulukları:**

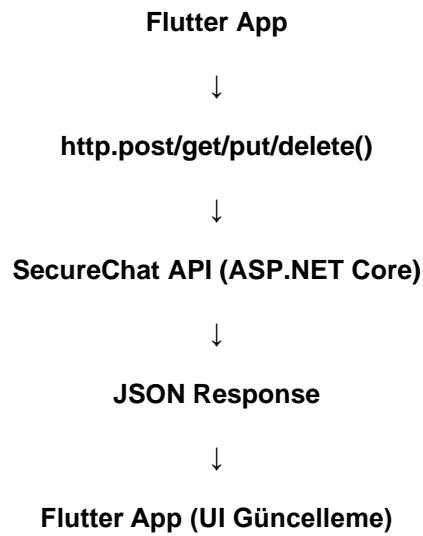
- Veri güvenliği ve bütünlüğü
- ACID özelliklerini sağlama

- Performans optimizasyonu (indexing)
- Yedekleme ve recovery

## 2.3. Bileşenler Arası İletişim

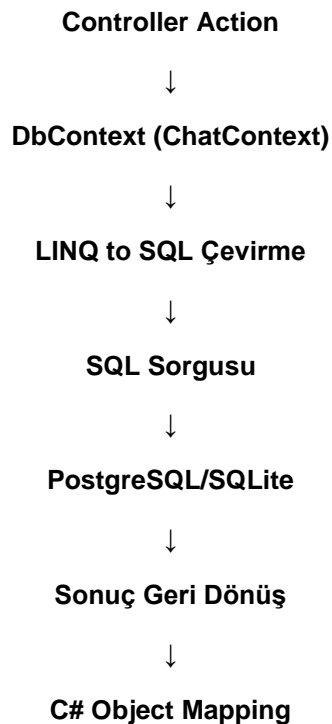
Frontend → Backend (HTTP REST API)

İstek Akışı:



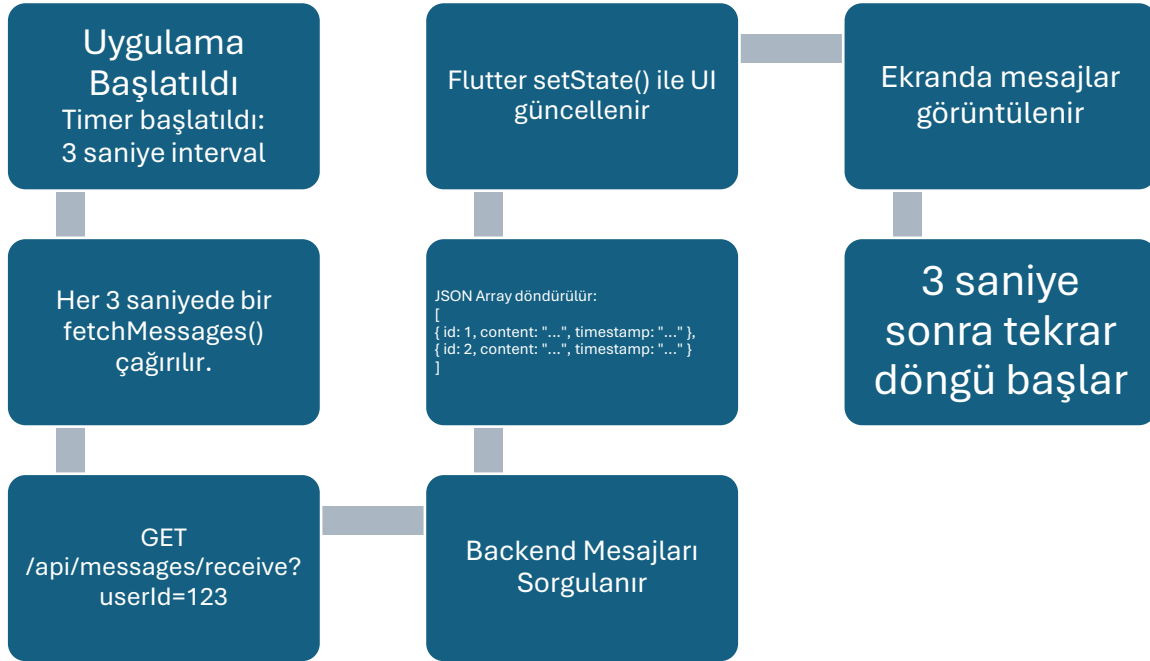
Backend ↔ Database (Entity Framework Core)

Sorgu Akışı:



## 2.4. Veri Akışı Döngüsü (Polling Mekanizması)

SecureChat, gerçek zamanlı mesaj güncellemeleri için HTTP Polling kullanır:



### 3. Veritabanı Tasarımı

Veritabanı olarak SQLite seçtim. Çünkü geliştirme sırasında kurulumu kolay, ek bir sunucu gerektirmeyen, tüm verileri tek bir dosyada tutulan bir sistemdir. Verileri organize etmek için tabloları kullanıcılar, mesajlar, gruplar gibi mantıklı parçalara bölüp, aralarında bağlantı oluşturdum. Böylece aynı bilgiyi tekrar yazmak yerine, bir yerde tutup referans gösteriyorum.

#### 3.1. Tablolar

Tablo	Açıklama
<b>Users</b>	Kullanıcı hesapları (ad, email, şifre, profil fotoğrafı)
<b>Messages</b>	Gönderilen ve alınan mesajlar
<b>Groups</b>	Grup sohbetleri
<b>GroupMembers</b>	Hangi kullanıcının hangi gruba üye olduğu
<b>MessageReactions</b>	Mesajlara eklenen emojiiler
<b>Status</b>	Kullanıcıların çevrimiçi mi çevrimdışı mı olduğu
<b>Notifications</b>	Bildirimler



## 4. Backend API Tasarımı

Backend'de .NET Core kullanarak REST API oluşturdum. Her işlem için (kullanıcı giriş, mesaj gönderme, grup oluşturma vb.) farklı endpoint'ler tasarladım. Veriler JSON formatında Frontend'e geri gönderilir.

### 4.1. Ana API Endpoint'leri

Kullanıcı İşlemleri (UserController)

- POST /api/user/register - Yeni kullanıcı kaydet
- POST /api/user/login - Kullanıcı giriş
- GET /api/user/{userId} - Kullanıcı bilgileri getir
- PUT /api/user/update - Profil güncelle
- GET /api/user/search?keyword= - Kullanıcı ara

Mesaj İşlemleri (MessagesController)

- POST /api/messages/send - Mesaj gönder
- GET /api/messages/receive?userId= - Mesajları al
- PUT /api/messages/{messageId} - Mesajı düzenle
- DELETE /api/messages/{messageId} - Mesajı sil
- POST /api/messages/{messageId}/react - Emoji tepki ekle

Grup İşlemleri (GroupsController)

- POST /api/groups/create - Yeni grup oluştur
- GET /api/groups/{userId} - Kullanıcının gruplarını getir
- POST /api/groups/{groupId}/addmember - Üye ekle
- DELETE /api/groups/{groupId}/removemember - Üye çıkar

Ayarlar & Durum (SettingsController, StatusController)

- PUT /api/settings/language - Dil değişimi
- POST /api/status/update - Son görülme güncelle
- GET /api/status/{userId} - Kullanıcı durumu al

## 5. Frontend Tasarımı

Flutter ile kullanıcı arayüzü oluşturdum. Material Design 3 kullanarak modern ve responsive tasarımlar yaptım. Uygulamanın farklı ekranları (login, chat, profil vb.) ayrı dart dosyalarında organize edilmiştir.

### 5.1. Ana API Endpoint'leri

Ekran	Dosya	Açıklama
Giriş Ekranı	login_page.dart	Kullanıcı kaydı ve giriş
Sohbet Ekranı	chat_page.dart	Ana mesajlaşma ekranı, mesaj gönderme/alma
Grup Ekranı	group_pages.dart	Grup oluşturma ve yönetimi
Profil Ekranı	profile_page.dart	Kullanıcı profili, ayarlar, dil değişimi

### 5.2. Ana Bileşenler

- **TextField** - Mesaj yazma alanı
- **ListView** - Mesajların scrollable listesi
- **Image.memory** - Gönderilen fotoğrafları gösterme
- **ElevatedButton** - Gönder butonu
- **AppBar** - Üst navigasyon bar'ı
- **GestureDetector** - Mesajlara dokunma işlemleri (cevaplama, silme, emoji ekleme)

## 6. Uygulama Özellikleri ve İş Akışları

SecureChat uygulamasında kullanıcılar birbirlerine güvenli şekilde mesaj gönderebilir, gruplar oluşturabilir ve çeşitli işlemler yapabilir. Tüm veriler Backend API aracılığıyla sunucuda depolanır.

### 6.1. Temel Özellikler

Özellik	Açıklama
<b>Bir-bir Mesajlaşma</b>	İki kullanıcı arasında özel sohbet
<b>Grup Mesajlaşması</b>	Çok kişi ile aynı grupta sohbet
<b>Mesaj Gönderme/Alma</b>	3 saniye polling ile mesajlar güncellenir
<b>Medya Gönderme</b>	Resim ve dosya gönderme (base64 şifreleme)
<b>Cevaplama</b>	Belirli bir mesaja cevap verme (gri kutuda gösterilir)
<b>Emoji Reaksiyonlar</b>	Mesajlara emoji tepki ekleme
<b>Mesaj Silme</b>	"Benim için sil" veya "Herkesten sil" seçeneği
<b>Son Görülme</b>	Kullanıcının ne zaman çevrimiçi olduğu görünür
<b>Dil Desteği</b>	Türkçe ve İngilizce

### 6.2. Kullanıcı Akışları

#### Giriş Akışı:

1. Kullanıcı login\_page'de email ve şifre girer
2. "Giriş" butonuna basılır → Backend API'ye istek gönderilir
3. Backend şifreyi kontrol eder
4. Başarılı ise chat\_page'e yönlendirilir

#### Mesaj Gönderme Akışı:

1. Kullanıcı chat\_page'de mesaj yazar
2. "Gönder" butonuna basılır
3. Backend'e POST isteği gönderilir

4. Mesaj veritabanında kaydedilir
5. Alıcı 3 saniye sonra polling ile mesajı görür

**Resim Gönderme Akışı:**

1. Kullanıcı "Medya Gönder" butonuna basılır
2. Galeri açılır, resim seçilir
3. Resim base64 stringine çevrilir
4. Backend'e mesaj ile birlikte gönderilir
5. Veritabanında base64 string olarak kaydedilir
6. Alıcı mesajı aldığı anda base64 deşifre edilerek resim gösterilir

## 7. SONUÇ

SecureChat projesi, modern anlık mesajlaşma uygulamalarının temel özelliklerini içeren, Flutter ve .NET teknolojileri kullanılarak geliştirilmiş kapsamlı bir mobil mesajlaşma platformudur. Proje, istemci-sunucu mimarisi üzerine kurulmuş olup, Flutter ile geliştirilmiş mobil uygulama ve .NET 8.0 ile geliştirilmiş RESTful API backend'inden oluşmaktadır. SQLite veritabanı kullanılarak veri yönetimi sağlanmış ve Entity Framework Core ile veritabanı işlemleri gerçekleştirilmiştir.

Proje kapsamında, kullanıcılar arasında metin mesajlaşması, grup sohbetleri, medya paylaşımı (fotoğraf, video, dosya), durum paylaşımı, mesaj düzenleme ve iletme, emoji reaksiyonları gibi temel özellikler başarıyla uygulanmıştır. Ayrıca karanlık tema desteği, çoklu dil desteği (Türkçe/İngilizce), çevrimiçi durum gösterimi ve son görülme özellikleri gibi kullanıcı deneyimini artıran özellikler eklenmiştir. Gelişmiş grup yönetimi sayesinde kullanıcılar grup oluşturabilir, üye ekleyip çıkarabilir ve admin yetkileri atayabilirler.

Proje, modern yazılım geliştirme prensiplerine uygun olarak modüler bir yapıda tasarlanmıştır. Backend tarafında controller'lar, model'ler ve veritabanı migration'ları düzenli bir şekilde organize edilmiştir. Frontend tarafında ise ekranlar, yardımcı fonksiyonlar ve yerelleştirme dosyaları ayrı modüller halinde yapılandırılmıştır. CORS yapılandırması ile farklı platformlardan API'ye erişim sağlanmış ve Swagger entegrasyonu ile API dokümantasyonu oluşturulmuştur.

Gelecekte, push notification entegrasyonu, WebRTC ile sesli ve görüntülü arama özellikleri, gerçek end-to-end şifreleme (AES), SignalR ile gerçek zamanlı iletişim ve offline mesaj senkronizasyonu gibi gelişmiş özellikler eklenerek proje daha da zenginleştirilebilir. Mevcut yapı, bu tür özelliklerin kolayca entegre edilmesine olanak sağlayacak şekilde tasarlanmıştır. Proje, temel mesajlaşma ihtiyaçlarını karşılayan, genişletilebilir ve bakımı kolay bir platform olarak başarıyla tamamlanmıştır.