



**ToGather Mobil Uygulaması**

**BİTİRME PROJESİ FİNAL RAPORU**

Bilgisayar Mühendisliği Bölümü

**DANIŞMAN**

Doç. Dr. Önder Demir

**ÖNSÖZ**

Bu belge, ToGather Mobil Uygulaması projemizin belirli bir aşamasını kapsayan **final raporu** niteliğindedir. Proje sürecinde şimdiye kadar elde ettiğimiz gelişmeleri, karşılaştığımız sorunları ve bu sorunlara yönelik çözüm yaklaşımlarımızı sistematik bir şekilde sunmayı amaçlamaktadır.

Mobil uygulama geliştirme süreci boyunca hem teknik bilgi birikimimizi artırma hem de ekip içi işbirliği ve proje yönetimi konularında önemli deneyimler edinme fırsatı bulduk. Özellikle yazılım mimarisi, kullanıcı arayüzü tasarımı, veri yönetimi ve hata ayıklama konularında pratik bilgiler edinerek, teorik bilgimizi uygulama sürecine aktarma imkânı elde ettik.

Bu final raporu, projenin genel hedefleri doğrultusunda geldiğimiz noktayı değerlendirmek ve bundan sonraki geliştirme sürecine ışık tutmak amacıyla hazırlanmıştır. Sürecin devamında edineceğimiz yeni bilgiler ve deneyimlerle projemizi daha da geliştirerek başarılı bir final ürün ortaya koymayı hedeflemekteyiz.

.

**İÇİNDEKİLER**

[**1.** **GİRİŞ**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.ythaxr7te570) **6**

[**1.1.** **Proje Konusu Ve Amacı**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.7moo9rbgq3d) **6**

[**1.2.** **Proje Kapsamı**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.7moo9rbgq3d) **7**

[**1.3.** **Kullanılan Yöntem ve Teknolojiler**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.7moo9rbgq3d) **8**

**2**[**.**](#_kng5gde7c5s) **GELİŞTİRME SÜRECİNE DAİR BULGULAR 10**

**2**[**.1.** **Öğrenilen Yeni Teknolojiler Ve Yaklaşımlar**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.63qsmftri2in) **10**

**2**[**.2.** **Karşılaşılan Sorunlar Ve Çözümler**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.63qsmftri2in) **11**

**3**[**.** **MEVCUT DURUM VE DEĞERLENDİRME**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.rucfmxu5z7r) **15**

**3**[**.1.** **Tamamlanan Kısımlar**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.63qsmftri2in) **15**

**3.2.** [**Devam Eden Ve Planlanan Geliştirmeler**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.63qsmftri2in) **18**

**4**[**.** **BULGULAR VE SONUÇ**](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.rucfmxu5z7r) **20**

**ÖZET**

**ToGather Mobil Uygulaması**

ToGather, insanların bir araya gelmesini kolaylaştırmak amacıyla geliştirilen bir mobil uygulamadır. Kullanıcılar uygulama üzerinden gruplar oluşturabilir, bu gruplara katılabilir ve ortak buluşma noktaları belirleyebilir. Harita üzerinden konum paylaşımı yapılabilir ve daha önce gerçekleşen buluşmalar geçmişte görüntülenebilir. Uygulama, sosyal etkileşimi kolaylaştırmayı ve planlama sürecini herkes için daha pratik hâle getirmeyi amaçlamaktadır.

**Haziran, 2025 Öğrenciler**

**Enes EKŞİ - Oğuz EVİRGEN** 

**ABSTRACT**

**ToGather Mobile Application**

ToGather is a mobile application developed to make it easier for people to come together. Through the app, users can create groups, join existing ones, and decide on common meeting points. Location sharing is available via an integrated map, and past gatherings can be viewed in a history section. The main goal of the app is to simplify social interactions and make the planning process more practical for everyone.

**June, 2025 Students**

**Enes EKŞİ - Oğuz EVİRGEN**



**KISALTMALAR**

**GMS :** Google Maps SDK

**CI/CD :** Continuous Integration / Continuous Deployment

**MVI :** Model–View–Intent

**DB**  : Database

**UI** : User Interface

**CRUD :** Create, Read, Update, Delete

**ŞEKİL LİSTESİ**

[**Şekil 1.1** Harita üzerinde kullanıcıların konumuna uygun avatarlarının bulunduğu mobil arayüz tasarımı.](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.gi4rp3xvou82)

## **1. GİRİŞ**

### **1.1. Proje Konusu ve Amacı**

Bu proje kapsamında geliştirilen **ToGather**, kullanıcıların grup oluşturarak ortak buluşma planları yapabildiği, harita üzerinden konum seçimi gerçekleştirebildiği ve buluşma geçmişlerini takip edebildiği, Android tabanlı bir mobil uygulamadır. ToGather uygulaması, bireysel kullanıcıların yanı sıra ilerleyen aşamalarda **daha geniş çaplı sosyal organizasyonlara** da ev sahipliği yapabilecek şekilde ölçeklenebilir olarak tasarlanmıştır.

Uygulamanın temel amacı, kullanıcıların sosyal etkileşimlerini organize etmelerini kolaylaştırmak ve ortak karar alma süreçlerinde yaşanan karışıklıkları ortadan kaldırmaktır. Bu kapsamda kullanıcılar, kendi gruplarını oluşturabilir, grup üyeleriyle birlikte buluşma noktası belirleyebilir ve planlamalarını harita entegrasyonu üzerinden kolaylıkla gerçekleştirebilir.

**ToGather**, yalnızca bireyler arası etkileşimle sınırlı kalmayıp, aşağıdaki gelişmiş özellikleri de desteklemeyi hedeflemektedir:

* **Daha büyük çaplı etkinlik organizasyonları**: Topluluklar, kulüpler veya işletmelerin geniş katılımlı etkinlikler planlayabileceği organizasyon modülü,
* **Konum bazlı kampanya ve fırsatlar**: Uygulama üzerinden kullanıcıların bulundukları konuma yakın **k**afeler, restoranlar veya etkinlik mekânlarının özel kampanyalarını görüntüleyebilmesi ve bu kampanyalara kolayca katılım sağlayabilmesi,
* **İşletmelere özel tanıtım alanları**: İşletmelerin uygulama üzerinden kullanıcı kitlesine ulaşabileceği, promosyon veya duyurular yapabileceği yapılar,
* **Etkinlik daveti ve katılım sistemi**: Kullanıcıların davet gönderip alabileceği, katılım durumu belirtme sistemlerinin desteklenmesi.

Bu genişletilebilir yapı sayesinde **ToGather** yalnızca bir buluşma planlama uygulaması değil, aynı zamanda sosyal etkileşimi teşvik eden, kullanıcıları çevresel fırsatlarla buluşturan ve yerel işletmeleri dijital ortama taşıyan bir platform haline gelecektir.

Uygulamanın ilk fazında, temel işlevsellik olan grup oluşturma, harita entegrasyonu ve buluşma geçmişi yönetimi üzerinde çalışılmıştır. Ancak tüm yazılım mimarisi, gelecekte bu vizyonun gerçekleşebilmesine olanak sağlayacak şekilde esnek ve sürdürülebilir olarak kurgulanmıştır.

### **1.2. Kapsam**

Modern yaşamda bireylerin sosyal etkileşimlerini planlamak ve sürdürmek, zaman ve mekân koordinasyonu açısından çeşitli zorluklar barındırmaktadır. Özellikle arkadaş grupları, kulüpler, topluluklar veya öğrenci grupları gibi küçük grupların bir araya gelme süreçlerinde; tüm üyelerin uygunluk durumu, buluşma yeri seçimi ve bu bilgilerin paylaşılması ciddi zaman kaybına ve karışıklığa neden olabilmektedir.

ToGather, bu soruna yönelik pratik ve kullanıcı dostu bir çözüm sunmaktadır. Özellikle konum bazlı buluşma noktası belirleme, grup oluşturma, bildirimler, etkinlik geçmişi ve konum izinlerine bağlı harita takibi gibi işlevleriyle, kullanıcıların daha hızlı ve etkili bir şekilde plan yapmalarını sağlar. Ayrıca mobil platformda geliştirilmiş olması, uygulamanın taşınabilirliğini artırmakta ve kullanıcıların istedikleri an uygulamaya erişmelerine olanak tanımaktadır.

Bu proje aynı zamanda, Android tabanlı uygulama geliştirme süreci boyunca öğrencilerin yazılım mimarisi, kullanıcı arayüzü tasarımı, veri yönetimi ve üçüncü parti API entegrasyonları konularında uygulamalı bilgi ve beceri kazanmalarını sağlamaktadır. Bu yönüyle ToGather, yalnızca teknik bir ürün değil, aynı zamanda öğrenme temelli bir gelişim projesi olarak da değerlendirilmektedir.

### 

### **1.3. KULLANILAN YÖNTEM VE TEKNOLOJİLER**

ToGather mobil uygulaması, güncel yazılım geliştirme yaklaşımları ve modern teknolojiler temel alınarak iki ana bileşen üzerinde yapılandırılmıştır: **Android (mobil istemci)** ve **Backend (sunucu tarafı)**. Her iki katman da yüksek okunabilirlik, sürdürülebilirlik, test edilebilirlik ve genişletilebilirlik prensipleri doğrultusunda geliştirilmiştir.

#### **1.3.1. Android Tarafı (Mobil Uygulama)**

Android uygulaması, **Kotlin** programlama dili ile geliştirilmekte olup, güncel mimari yaklaşımlardan biri olan **MVI (Model–View–Intent)** yapısı temel alınmıştır. MVI mimarisi sayesinde kullanıcı arayüzü ile veri katmanı arasındaki ilişki tek yönlü veri akışıyla yönetilmekte ve UI durumları (State) merkezi bir yapı üzerinden kontrol edilmektedir. Bu yaklaşım, özellikle hata ayıklama, test edilebilirlik ve yeniden kullanılabilirlik açısından önemli avantajlar sağlamaktadır.

Mobil uygulama geliştirme sürecinde kullanılan başlıca teknolojiler ve kütüphaneler şunlardır:

* **Hilt:** Dependency Injection (bağımlılık enjeksiyonu) yapısını sağlamak için kullanılmıştır. Uygulamanın modülerliğini ve test edilebilirliğini artırmaktadır.
* **Retrofit:** API isteklerinin yönetilmesi ve RESTful servislerle haberleşme için tercih edilmiştir.
* **Room:** Yerel veritabanı işlemlerinin yönetilmesinde kullanılır. Uygulama içerisinde bazı verilerin çevrimdışı erişilebilirliği için tercih edilmektedir.
* **Google Maps SDK (GMS):** Harita gösterimi ve konum tabanlı işlemler (buluşma noktası belirleme vb.) için entegre edilmiştir.
* **Firebase:** Kimlik doğrulama, bildirim gönderimi ve temel veri saklama işlemleri için arka plan servis olarak kullanılmaktadır.
* **Coroutines & StateFlow:** Asenkron işlemler ve UI durum yönetimi için Kotlin’in coroutine yapısı ile birlikte StateFlow kullanılmaktadır.

#### **1.3.2. Backend Tarafı**

Uygulamanın Backend tarafı, **Java 21** ile yazılmış ve **Spring Boot 3.4.4** çerçevesi ile yapılandırılmıştır. Backend sistemi, mobil istemciden gelen talepleri karşılamak, veri yönetimini gerçekleştirmek ve kullanıcı işlemlerini (grup oluşturma, buluşma geçmişi kaydı, kampanya entegrasyonu vb.) yürütmek üzere tasarlanmıştır.

Kullanılan başlıca Backend tarafı teknolojiler şunlardır:

* **Spring Boot 3.4.4:** Mikroservis mimarisi prensiplerine uygun, hafif ve genişletilebilir bir backend altyapısı sağlar.
* **Docker:** Uygulamanın konteynerize edilmesi sayesinde platform bağımsız olarak taşınabilirlik ve kolay dağıtım sağlanmıştır.
* **Spring Security:** API katmanında temel kimlik doğrulama ve yetkilendirme işlemleri için kullanılmaktadır.
* **Deployment Teknolojileri:** Projenin ilerleyen aşamalarında **CI/CD (Continuous Integration / Continuous Deployment)** süreçleri entegre edilerek, otomatik test ve dağıtım süreçleri hayata geçirilecektir. Bulut ortamına deploy için çeşitli alternatifler değerlendirilmektedir (örneğin: AWS, Google Cloud, Railway.app vb.).
* **Veritabanı:** Verilerin kalıcı olarak saklanması için ilişkisel veritabanı sistemleri tercih edilmektedir. Bu kapsamda PostgreSQL veritabanı ile çalışılmıştır.

## **2. GELİŞTİRME SÜRECİNE DAİR BULGULAR**

### **2.1. Öğrenilen Yeni Teknolojiler ve Yaklaşımlar**

ToGather projesi kapsamında yürütülen geliştirme süreci, sadece uygulama ortaya koyma amacıyla değil, aynı zamanda teknik bilgi birikimini artırma ve yeni teknolojileri uygulamalı olarak öğrenme açısından da oldukça verimli olmuştur. Proje boyunca hem mobil hem de backend tarafında birçok yeni teknolojiyle doğrudan çalışma fırsatı yakalanmıştır. Bu süreçte edinilen başlıca kazanımlar aşağıda iki ana başlık altında sunulmuştur.

#### **2.1.1. Android Tarafında Edinilen Bilgi ve Beceriler**

Mobil uygulama geliştirme sürecinde daha önce temel düzeyde bilgi sahibi olunan bazı konular derinlemesine öğrenilmiş ve yeni teknolojiler ilk kez deneyimlenmiştir:

* **MVI (Model-View-Intent) Mimarisi**: Proje süreci boyunca MVI mimarisi uygulamalı şekilde kullanılarak bu yapının temel bileşenleri olan ViewState, Intent ve Reducer gibi kavramlar detaylı olarak kavranmıştır. Tek yönlü veri akışının nasıl sağlandığı, kullanıcı arayüzü durumlarının nasıl yönetildiği ve geri dönülebilir UI durumları oluşturmanın önemi gibi konularda pratik bilgi edinilmiştir.
* **Clean Code Prensipleri:** Kodun okunabilirliği, sürdürülebilirliği ve test edilebilirliği açısından önem arz eden temiz kod prensipleri, modüler yapı, fonksiyonel ayrım, sadeleştirme, anlamlı isimlendirme ve bağımlılıkların netleştirilmesi gibi kriterler üzerinden uygulamalı olarak öğrenilmiştir.
* **Google Maps SDK (GMS) Kullanımı:** İlk kez aktif bir harita bileşeniyle çalışılmış; harita gösterimi, kullanıcı konumu alma, marker yerleştirme ve konum izinlerinin yönetimi gibi fonksiyonlar üzerinden GMS API'lerinin nasıl entegre edildiği ve yönetildiği öğrenilmiştir.

#### **2.1.2. Backend Tarafında Edinilen Bilgi ve Beceriler**

Sunucu tarafında yapılan geliştirmeler sayesinde backend teknolojileriyle ilgili birçok yeni kavram ve yapı ilk kez deneyim edilmiştir. Özellikle büyük ölçekli ve üretime uygun bir sunucu uygulamasının nasıl kurgulanacağına dair değerli deneyimler elde edilmiştir:

* **CI/CD Süreçleri:** Uygulamanın geliştirme ve dağıtım süreçlerinin otomatikleştirilmesi için kullanılan sürekli entegrasyon ve sürekli dağıtım (CI/CD) prensipleri öğrenilmiştir. Bu sayede hem kod kalitesi hem de üretim ortamına geçiş süreçleri daha güvenli ve hızlı hâle gelmektedir.
* **Container ve Sunucu Mantığı:** Docker kullanımı ile birlikte uygulamanın platform bağımsız olarak çalıştırılabilmesi sağlanmış, container mimarisi ile yazılımın taşınabilirliği ve dağıtımı konusunda pratik bilgi edinilmiştir. Uygulamanın fiziksel bir makine yerine izole bir ortamda çalışmasının avantajları doğrudan deneyimlenmiştir.
* **Core Katman ve Bağımlılık Yönetimi:** Spring Boot çerçevesinde projeyi modüler yapıda kurmak amacıyla core katmanlarının ayrılması, servis, veri ve yapı katmanlarının izole edilmesi ve farklı bileşenlerin birbiriyle olan bağımlılıklarının doğru şekilde yönetilmesi konularında tecrübe kazanılmıştır. Bu sayede daha okunabilir, test edilebilir ve genişletilebilir bir backend yapısı oluşturulmuştur.

### **2.2. Karşılaşılan Sorunlar Ve Çözümler**

**2.2.1. Grup Detay Ekranı Akışı Hataları**  
Grup Ekranı Tasarım Eksiklikleri: Grup oluşturma ve yönetim ekranlarının ilk tasarımlarında kullanıcı deneyimi açısından eksiklikler olduğu gözlemlenmiştir. Arayüzün karmaşık olması ve tutarsız görsel unsurlar, kullanıcıların grup işlemlerini gerçekleştirirken zorluk yaşamalarına neden olmuştur. Bu sorunu çözmek için uygulamanın arayüz tasarımı, Google Material Design yönergelerine uygun olarak yeniden ele alınmıştır. Material Design kılavuzları, platformlar arası tutarlı ve kullanıcı dostu arayüzler oluşturmak için kapsamlı prensipler sunmaktadır. Bu doğrultuda renk paleti, tipografi ve bileşen düzenleri tutarlı hale getirilmiş; gezinti menüleri ve buton konumları kullanıcı alışkanlıklarına uygun şekilde düzenlenmiştir. Sonuç olarak grup ekranları, daha sade bir tasarım ve anlaşılır bir düzen ile yeniden geliştirilerek kullanıcı deneyimi iyileştirilmiştir.

#### **2.2.2. Tasarım Süreci ve Arayüz Oluşturma Problemleri**

Projenin başlangıç aşamasında, kullanıcı arayüzü (UI) tasarımlarının hazırlanması sürecinde birtakım zorluklar yaşanmıştır. Özellikle Figma gibi profesyonel tasarım araçlarına hâkimiyet eksikliği, tutarlı ve kullanıcı dostu bir arayüz oluşturulmasını zorlaştırmıştır. Görsel uyumsuzluklar, boşluk ve hizalama problemleri ile sezgisel olmayan ekran akışları gibi sorunlarla karşılaşılmıştır.

Bu problemi çözmek adına, Google tarafından sunulan **Material Design** yönergeleri detaylı olarak incelenmiş ve Android Jetpack’in sunduğu **Material Components** aktif olarak kullanılmaya başlanmıştır. Bu sayede tasarım süreçleri daha sistematik ve standartlara uygun şekilde ilerlemiş; düğmeler, kartlar, formlar ve gezinme bileşenleri gibi birçok öğe için hazır çözümler uygulanarak modern ve tutarlı bir kullanıcı arayüzü elde edilmiştir.

#### **2.2.3. Backend ve Android Client Arasındaki Bağlantı Problemleri**

Proje sürecinde mobil uygulama ile sunucu arasında veri alışverişi yapılması planlandığında, backend servislerinin henüz canlıya (production ortamına) alınmamış olması çeşitli entegrasyon sorunlarına neden olmuştur. Özellikle Android istemciden gelen taleplerin cevaplanamaması ve lokal ortamda çalışan backend servislerine mobil cihazdan doğrudan erişim sağlanamaması önemli bir engel oluşturmuştur.

Bu problem, backend uygulamasının **Docker** ortamında çalıştırılması ve hem yerel hem de mobil cihaz üzerinden erişilebilir hale getirilmesiyle aşılmıştır. Docker konteynerleri sayesinde uygulama bileşenleri izole bir ortamda çalıştırılmış; API testleri sağlıklı şekilde gerçekleştirilmiştir.

#### **2.2.4. Mimari Yapılanma ve Kod Organizasyonu Sorunları**

Projenin ilk aşamalarında, Android uygulamasının mimari yapısını kurgularken **ekranlar arası veri paylaşımı**, **durum yönetimi**, **modülerlik** ve **bağımlılıkların yönetimi** konularında çeşitli mimari karmaşalarla karşılaşılmıştır. Uygulamanın büyüdükçe yönetilemez hale gelmemesi için sürdürülebilir ve temiz bir yapı kurma gerekliliği doğmuştur.

Bu bağlamda, farklı açık kaynak projeler ve kurumsal uygulama örnekleri incelenmiş; MVI mimarisi esas alınarak yeni bir yapılanma oluşturulmuştur. Kodun sorumluluklara göre ayrılması (presentation, domain, data), dependency injection altyapısının Hilt ile yönetilmesi ve durum yönetiminin StateFlow yapısıyla ele alınması sayesinde bu problemler aşılmıştır.

#### **2.2.5. Google Maps SDK Kullanımı ile İlgili Zorluklar**

Uygulamanın önemli işlevlerinden biri olan harita tabanlı buluşma noktası belirleme sürecinde, **Google Maps SDK** entegrasyonu sırasında teknik zorluklarla karşılaşılmıştır. Özellikle konum izinlerinin yönetimi, harita üzerinde marker yerleştirme ve kullanıcıya en iyi deneyimi sunacak harita konumlamasının sağlanması konularında çeşitli denemeler yapılmıştır.

Bu sorunlar halen proje kapsamında üzerinde çalışılmakta olan bir alan olup, daha stabil ve kullanıcı dostu bir harita deneyimi oluşturmak amacıyla araştırmalar ve denemeler devam etmektedir. Bu bağlamda, Google Maps’in resmi dökümantasyonu, Stack Overflow tartışmaları ve çeşitli açık kaynak projelerden faydalanılmaktadır. Uygulamada kullanıcılar ve etkinlikler harita üzerinde görüntülenebilmektedir. Projenin ilerleyen safhalarında harita üzerinde çok sayıda konum işaretçisi (marker) gösterilmesi gereksinimi ortaya çıkmıştır. Ancak, aynı anda çok sayıda marker göstermek performans sorunlarına ve harita arayüzünde kalabalık bir görüntüye yol açmıştır. Bu sorunu aşmak için Google Maps SDK’nin marker clustering özelliği kullanılmıştır. Marker clustering, harita üzerinde çok sayıda işaretçi olduğunda bunları gruplandırarak göstermeyi ve böylece hem performansı hem de okunabilirliği artırmayı sağlar. Uygulamada yakın konumdaki noktalar belirli bir uzaklaştırma seviyesinde tek bir küme (cluster) ikonu olarak gösterilecek şekilde implementasyon yapılmıştır. Bu sayede harita etkileşimleri (kaydırma, yakınlaştırma) daha akıcı hale getirilmiş ve kullanıcının harita üzerindeki veriyi algılaması kolaylaştırılmıştır. Ayrıca, harita optimizasyonu kapsamında, harita hareket halindeyken yeni verilerin yüklenmemesi ve gereksiz yeniden çizimlerin önlenmesi gibi Google Maps Platformu en iyi uygulamalarına da dikkat edilmiştir. Bu iyileştirmeler sonucunda uygulamanın harita modülü, yüksek sayıda veri noktasını dahi verimli şekilde yönetebilen bir hale gelmiştir.

#### **2.2.6. Firebase Analytics Entegrasyon Problemleri**

Firebase Analytics Entegrasyonu: Uygulamanın son aşamasında, kullanıcı davranışlarının ve etkileşimlerinin izlenebilmesi amacıyla Firebase Analytics (Google Analytics for Firebase) entegrasyonu gerçekleştirilmek istenmiştir. Entegrasyon sürecinde, Firebase SDK'sının projeye eklenmesi ve olayların doğru biçimde loglanması konusunda bazı zorluklar yaşanmıştır. Özellikle, Analytics servisinin etkin hale getirilmesi, google-services.json yapılandırması ve uygulama etkinliklerinden anlamlı olay verilerinin toplanması konularında dikkatli bir kurulum gerekmiştir. Bu sorun, Firebase dokümantasyonundaki adımların izlenmesi ve uygulamanın debug modunda Firebase DebugView aracı ile test edilmesiyle çözülmüştür. Sonuçta Firebase Analytics başarıyla entegre edilerek uygulamanın kullanım ve davranış verileri gerçek zamanlı olarak toplanmıştır. Google Analytics for Firebase, mobil uygulamanın kullanım sıklığı, popüler özellikleri ve kullanıcı demografisi gibi verileri olaylar ve kullanıcı özellikleri biçiminde toplayabilmektedir. Bu sayede geliştiriciler, uygulamanın hangi bölümlerinin daha fazla kullanıldığını objektif verilere dayanarak analiz etme ve gelecekteki iyileştirmeleri planlama imkanına kavuşmuştur.

## 

## 

## 

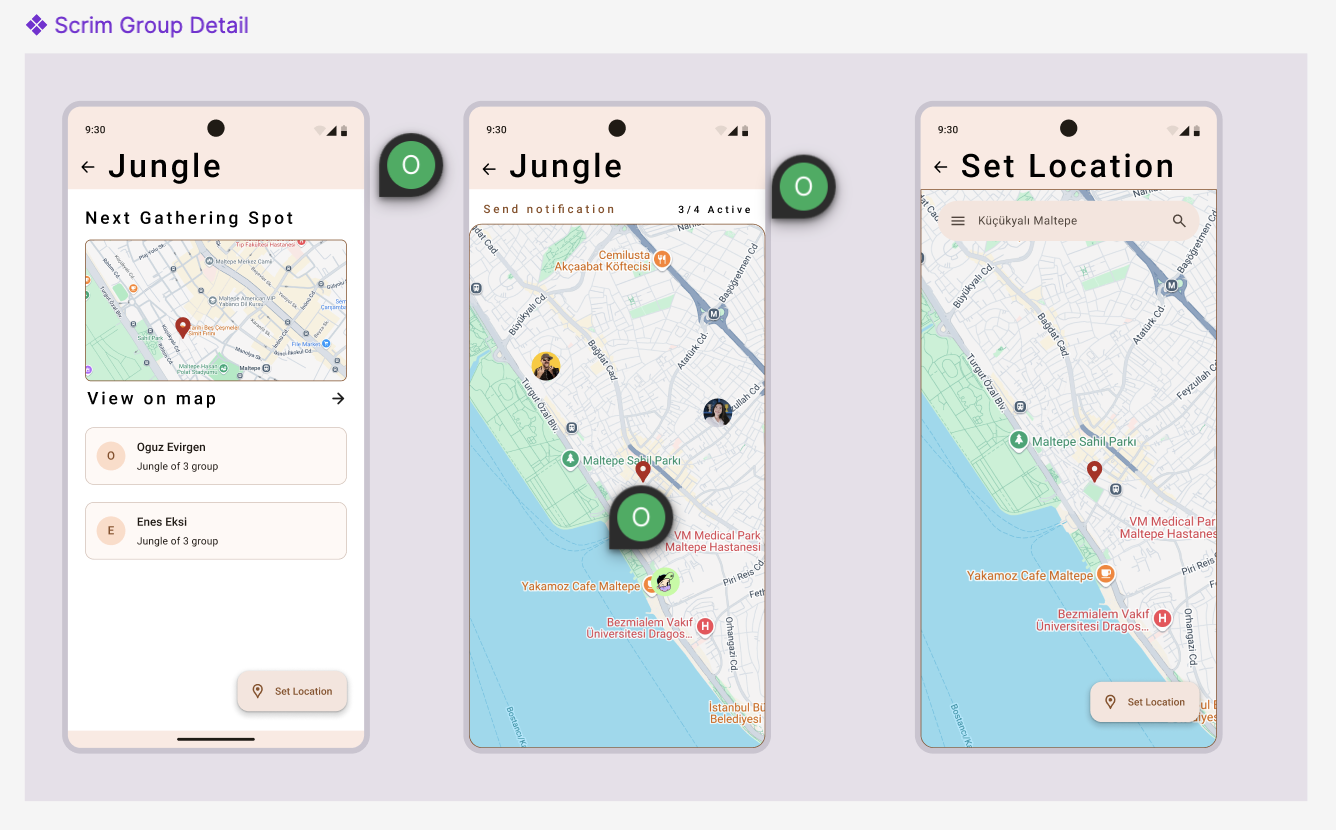
## 

## **3. MEVCUT DURUM VE DEĞERLENDİRME**

### **3.1. Tamamlanan Kısımlar**

#### **3.1.1. Arayüz Tasarımı ve Kullanıcı Deneyimi**

Uygulamanın arayüz tasarım süreci, kullanıcı deneyimini ön planda tutacak şekilde **Figma** aracı kullanılarak yürütülmüştür. Başlangıçta tasarıma yönelik deneyim eksikliği nedeniyle bazı zorluklar yaşansa da, bu süreçte Google’ın sunduğu **Material Design** ilkeleri incelenmiş ve rehber alınarak tasarım süreci yapılandırılmıştır.



**Şekil 1.1** [Harita üzerinde kullanıcıların konumuna uygun avatarlarının bulunduğu mobil arayüz tasarımı](https://docs.google.com/document/d/1RzKmsA3uxNbFtMEjml8NfNicYRDERDDm7N3yQ52k1-o/edit#heading=h.gi4rp3xvou82)

Material Design sistemine ait **hazır bileşenler (components)** kullanılarak oluşturulan ekranlarda tutarlı ve modern bir kullanıcı arayüzü elde edilmiştir. Örneğin, kullanıcı avatarları, harita kartları, konum belirleme düğmeleri ve gezinme unsurları gibi öğeler, Material Design bileşenleriyle inşa edilmiştir. Şekil 1.1’de paylaşılan ekran görsellerinde de görüldüğü üzere, grup detay ekranı, harita ekranı ve konum belirleme ekranı gibi temel UI bileşenleri tamamlanmıştır. Bu sayede kullanıcılar buluşma yerlerini görebilmekte, üyeleri listeleyebilmekte ve konum belirleme işlemlerini gerçekleştirebilmektedir.

#### **3.1.2. Backend Geliştirmeleri**

Sunucu tarafında geliştirilen modüller ve sistem altyapısı şu şekilde özetlenebilir:

* **Veritabanı Tasarımı (DB Design):** Kullanıcılar, gruplar, buluşma noktaları ve bildirim sistemini kapsayan temel veritabanı tabloları tasarlanmış ve ilişkisel model kurulmuştur.
* **Base Mimari Yapı:** Spring Boot çatısı altında sürdürülebilir ve modüler bir backend mimarisi kurulmuştur.
* **Database Bağlantıları:** Veritabanı işlemleri için JPA/Hibernate tabanlı yapı kurulmuş, repository katmanları oluşturulmuştur.
* **CI/CD Süreçleri:** Kodun sürümlenmesi, test edilmesi ve otomatik dağıtımı için temel continuous integration/deployment altyapısı oluşturulmuştur.
* **Kimlik Doğrulama ve Yetkilendirme:** JWT tabanlı Authentication ve kullanıcı bazlı Authorization mekanizmaları devreye alınmıştır.
* **CRUD Operasyonları:** Kullanıcı, grup ve buluşma noktaları gibi temel varlıklar için CRUD işlemlerini kapsayan REST servisleri geliştirilmiştir.
* **Exception Yönetimi:** Sunucu taraflı hata kontrolü için özelleştirilmiş global exception handler yapısı oluşturulmuştur.
* **Logger Yapısı:** Uygulamanın hata ve işlem kayıtlarının izlenebilmesi için merkezi bir logger sistemi yapılandırılmıştır.

#### 

#### 

#### 

#### **3.1.3. Android (Client) Geliştirmeleri**

Mobil uygulama tarafında tamamlanan başlıca yapılar şu şekildedir:

* **Base Mimarinin Kurulması:** Uygulama, MVI (Model–View–Intent) mimarisi üzerine kurulmuştur. ViewState, Event, Intent ve Reducer yapılarıyla modüler ve test edilebilir bir akış oluşturulmuştur.
* **Modern Android Kütüphaneleri:** Hilt (DI), Retrofit (API), Room (yerel veri), Coil (görsel yükleme), Coroutine & StateFlow (durum yönetimi) gibi güncel kütüphaneler projeye entegre edilmiştir.
* **Harita İşlevleri:** Google Maps SDK entegrasyonu sağlanarak kullanıcı konumu, marker gösterimi ve konum seçimi gibi özellikler başarıyla devreye alınmıştır.
* **Session Management:** Kullanıcının oturum yönetimi ve verisinin güvenli taşınması için temel session yapıları kurgulanmıştır.
* **Önbellekleme Mekanizmaları:** Yerel verilerin hızlı erişimi için temel cache yapıları (örneğin Room kullanımı) devreye alınmış ve ileriye dönük genişletilebilir hale getirilmiştir.

### 

### 

### 

### 

### 

### 

### 

### 

### **3.2. Devam Eden ve Planlanan Geliştirmeler**

#### **3.2.1. Android Tarafında Devam Eden Geliştirmeler**

* **Performans Refaktörleri:** Uygulama büyüdükçe, kod yapısının sürdürülebilir kalabilmesi için refactoring (yeniden yapılandırma) çalışmaları planlanmaktadır. Özellikle UI bileşenleri, ViewModel yapıları, veri akış mekanizmaları ve dependency injection (bağımlılık enjeksiyonu) bölümleri sadeleştirilerek geliştirilebilir ve okunabilir bir kod tabanı oluşturulacaktır.
* **​Kullanıcı Deneyimi İyileştirmeleri:** Devam eden bir çalışma olarak, uygulamanın animasyonlar ve geçiş efektleriyle daha akıcı hale getirilmesi hedeflenmektedir. Material Design prensiplerine uygun şekilde mikro animasyonların eklenmesi ve ekran geçişlerinin iyileştirilmesiyle kullanıcı deneyimi zenginleştirilecektir. Bunun yanında, karanlık mod (dark mode) desteğinin eklenmesi planlanmaktadır; böylece kullanıcılar tercihlerine göre farklı tema seçeneklerini kullanabilecektir.
* **Platform ve Cihaz Desteği:** Android uygulamasının stabilitesini artırmak ve daha geniş bir kullanıcı kitlesine erişmek amacıyla, farklı Android sürümlerinde ve çeşitli cihaz ekran boyutlarında kapsamlı testler devam etmektedir. Özellikle düşük bellekli veya eski model cihazlarda performansı optimize etmek için çalışmalar sürmektedir. Uygulamanın iOS versiyonunun geliştirilmesi de uzun vadeli planlar arasında yer almakta olup, mevcut iş mantıkları ve IOS Design sisteme uygun şekilde tasarlanmış ekranlar ile Swift yazılım dili ile IOS geliştirmeleri düşünülmektedir.
* **Yeni Özellikler:** Gelecekte eklenmesi planlanan bazı özellikler bulunmaktadır. Örneğin, grup içi anlık mesajlaşma veya tartışma panosu gibi iletişim özelliklerinin uygulamaya dahil edilmesi öngörülmektedir. Bu sayede grup üyeleri uygulama içerisinden daha etkin iletişim kurabilecektir. Ayrıca, konum tabanlı anımsatıcılar (Geocoder ile belirli bir bölgeye girildiğinde bildirim gönderilmesi) ve takvim entegrasyonu (cihaz takvimine etkinlik ekleme) gibi kullanıcıların hayatını kolaylaştıracak fonksiyonlar planlama aşamasındadır.
* **Bildirim Entegrasyonu:** Push Servisleri sayesinde kullanıcıya gönderilecek bildirimlerin atılması sağlanmıştır.

#### 

#### 

#### **3.2.2. Backend Tarafında Devam Eden Geliştirmeler**

* **Sunucuya Deployment (Canlı Yayın) İşlemleri:** Projenin üretim ortamına taşınabilmesi adına, dockerize edilmiş backend servislerinin bir bulut sağlayıcıya kalıcı olarak deploy edilmesi planlanmaktadır. Bu işlem tamamlandığında, Android uygulaması doğrudan gerçek sunucular üzerinden hizmet alacak ve uygulama, dış dünyaya açık hâle gelecektir.
* **CI/CD Pipeline Kurulumu(Sunucu Bazlı)**
* **Kafka Kurulumu ve Asenkron İletişim Altyapısı:** Projenin ilerleyen fazlarında mikroservis mimarisine evrilebilmesi adına, mesajlaşma altyapısı olarak Apache Kafka kurulması hedeflenmektedir. Kafka, servisler arası iletişimi asenkron hâle getirerek sistemin ölçeklenebilirliğini ve esnekliğini artıracaktır. Bu yapı özellikle yüksek kullanıcı yoğunluğuna sahip senaryolarda sistemin dayanıklılığını güçlendirecektir.
* **API Genişletmeleri:** Yeni planlanan mobil özelliklere paralel olarak, sunucu API’lerinde de genişletmeler yapılacaktır. Örneğin, anlık mesajlaşma özelliği eklenirse, bu işlemleri destekleyecek yeni WebSocket tabanlı servisler veya REST uç noktaları geliştirilecektir. Benzer şekilde, takvim entegrasyonu veya harici servis bağlantıları (örneğin sosyal medya paylaşım entegrasyonu) gibi yenilikler için gereken sunucu tarafı desteği planlanmaktadır. API dokümantasyonu da (Swagger/ OpenAPI üzerinden) güncel tutulacak ve yeni eklenen servislerin istemci geliştiriciler tarafından kolaylıkla kullanılabilmesi sağlanacaktır.
* **Ölçeklenebilirlik ve Yük Testleri:** Backend için devam eden en önemli geliştirme alanlarından biri ölçeklenebilirliktir. Artan kullanıcı sayısına ve veri yüküne sistemin uyum sağlayabilmesi için yük testleri yapılmakta ve sonuçlara göre optimizasyonlar planlanmaktadır. İleride, mikroservis mimarisi gibi yaklaşımlar ya da bulut tabanlı otomatik ölçeklendirme (auto-scaling) çözümleri değerlendirilebilir. Mevcut monolitik yapının gerektiğinde modüler hale getirilerek farklı sunuculara dağıtılması (örn. kullanıcı servisi, grup yönetim servisi gibi ayrıştırmalar) planlanmaktadır.
* **Performans ve Kod Kalitesi İyileştirmeleri:** Kodun daha sade, anlaşılır ve bakımı kolay bir hâle gelmesi için hem istemci hem de sunucu tarafında refactoring süreci devam etmektedir. Özellikle redundant kodların temizlenmesi, servis katmanlarının sadeleştirilmesi ve ortak bileşenlerin soyutlanması gibi adımlar uygulanacaktır.
* **Bildirim Entegrasyonu:** Push Servisleri sayesinde kullanıcıya gönderilecek bildirimlerin atılması sağlanmıştır.

## **4. BULGULAR VE SONUÇ**

ToGather mobil uygulaması projesinin geliştirme süreci boyunca elde edilen teknik bulgular ve sonuçlar, proje ekibi için öğretici olmuştur. İlk olarak, doğru teknoloji ve araç seçiminin proje başarısındaki önemi görülmüştür. Android platformu ve Spring Boot çerçevesinin kullanımı, istemci-sunucu mimarisinin hızlı bir şekilde hayata geçirilmesini sağlamıştır. Spring Boot, minimum konfigürasyon ile çalıştırılabilir, bağımsız ve üretim seviyesinde uygulamalar geliştirmeyi kolaylaştıran bir çatı olarak beklentileri karşılamıştır . Bu sayede ekip, temel fonksiyonlara odaklanarak hızlıca ilerleyebilmiştir. Aynı zamanda, Firebase gibi hazır altyapı servislerinin (analitik araçları) entegrasyonu, birçok karmaşık özelliğin güvenilir bir şekilde uygulanmasına imkan tanımıştır. Bu entegrasyonların dokümantasyona uygun yapılmasının, ortaya çıkabilecek sorunları en aza indirdiği deneyimlenmiştir.   
İkinci olarak, altyapı güçlendirme ve ölçeklenebilirlik konularında önemli deneyimler kazanılmıştır. Proje ilerledikçe artan veri yükü ve kullanıcı etkileşimleri, sistemin performans sınırlarını belirlemede yardımcı olmuştur. Yapılan optimizasyonlar (örneğin, harita üzerinde clustering kullanımı, veritabanı sorgularının iyileştirilmesi, HTTP istek sayılarının azaltılması) uygulamanın hem istemci hem sunucu tarafında daha verimli çalışmasını sağlamıştır. Elde edilen bulgulardan biri, erken aşamada yapılan performans testlerinin ve kod incelemelerinin, ileride karşılaşılabilecek ölçek sorunlarını engellemede kritik rol oynadığıdır. Proje kapsamında uygulanan cache mekanizmaları, uygun veri yapılarının seçimi ve asenkron işlem yapısı, sistemin daha ağır yükleri kaldırabilmesini mümkün kılmıştır.

Üçüncü olarak, kullanıcı deneyimi (UX) odaklı geliştirme sürecin merkezinde yer almıştır. Grup ekranlarının yeniden tasarlanması ve Material Design ilkelerinin uygulanması sonucu, kullanıcı geri bildirimlerinde belirgin iyileşmeler görülmüştür. Bu durum, yazılım projelerinde tasarım ve kullanıcı deneyimi prensiplerine uygunluğun ne denli önemli olduğunu göstermiştir. Uygulamada tutarlılık, erişilebilirlik ve basitlik sağlayarak kullanıcı memnuniyetinin arttığı tespit edilmiştir. Ayrıca, uygulamaya entegre edilen Google Analytics for Firebase sayesinde, kullanıcı davranış verilerinin toplanması mümkün olmuş ve bu veriler ışığında özellik önceliklendirmesi yapılmıştır. Örneğin, hangi özelliklerin daha çok kullanıldığı verisi, gelecek sürümlerde kaynakların nereye odaklanacağı konusunda ekibe yol göstermiştir.   
Sonuç olarak, ToGather projesi belirtilen hedefleri büyük ölçüde gerçekleştirmiş ve teknik açıdan tatmin edici bir ürün ortaya koymuştur. Ekip olarak, modern mobil teknolojilerini kullanarak tam fonksiyonel bir sistem geliştirmiş, karşılaşılan sorunları çözerek ürünün kalitesini arttıracak adımlar atılmıştır. Elde edilen deneyimler, ileride benzer çok katmanlı (client-server) sistem geliştirmelerinde referans teşkil edecek niteliktedir. Bu Final raporu ile belgelenen süreç, yazılım geliştirme yaşam döngüsünün analiz, tasarım, implementasyon, test ve devreye alma aşamalarındaki bulguları ortaya koymakta; gelecekte yapılabilecek geliştirmelere ışık tutmaktadır.

**KAYNAKLAR**

* [Material Design for Android | Views | Android Developers](https://developer.android.com/develop/ui/views/theming/look-and-feel)
* [Optimization Guide | Google Maps Platform | Google for Developers](https://developers.google.com/maps/optimization-guide)
* [Google Maps Android Marker Clustering Utility | Maps SDK for Android | Google for Developers](https://developers.google.com/maps/documentation/android-sdk/utility/marker-clustering)
* [Get started with Google Analytics | Google Analytics for Firebase](https://firebase.google.com/docs/analytics/get-started)
* [Spring Boot](https://spring.io/projects/spring-boot/)