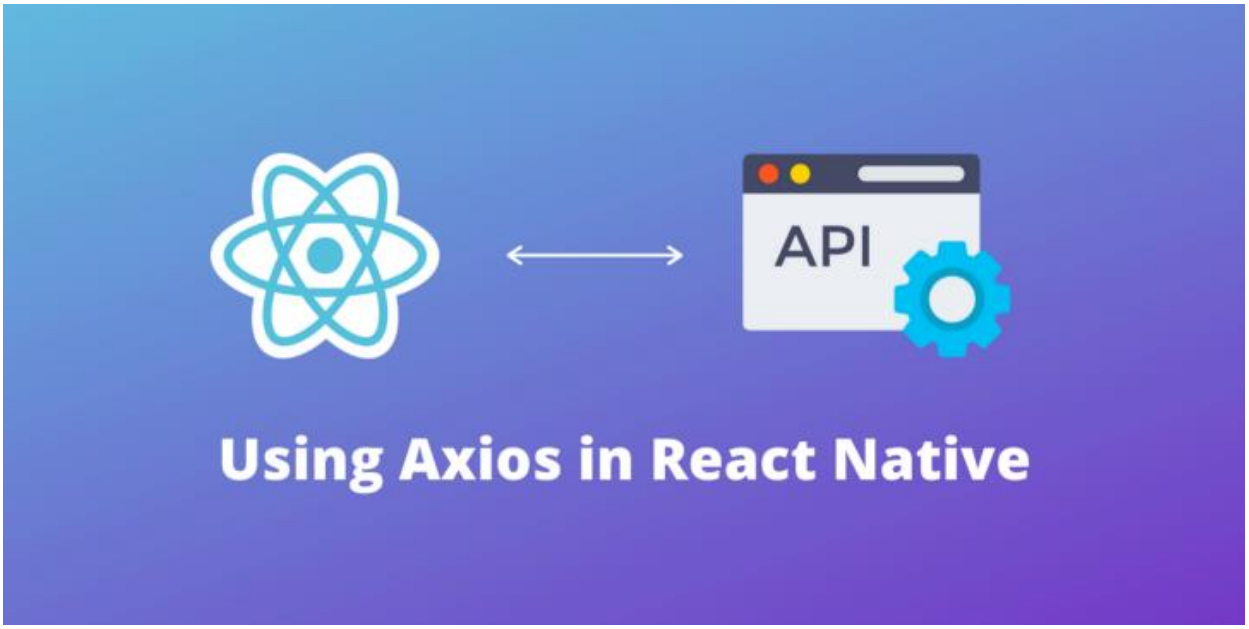
MİMARİ TASARIM DÖKÜMANI



Mimari olarak kısaca özetleyecek olursak yukarı da ki görsel ile anlatabiliriz uygulamamızı.

Ama daha detaylı bilgilere geçecek olursak; uygulamamızın back-end yapısını oluştururken public olan web API'leri kullanmayı tercih ettik. Çünkü Web API'lerden veri çekmek uygulama boyutu ve kullanılabilirliği açısından oldukça elverişliydi.

UI, User Interface, Kullanıcı Arayüzü Kiti kullanılarak uygulama ara yüzü tasarlama görevinde yardımcı olabilecek bir grafik dosyaları ve kaynak koleksiyonu sağlamış olduk. Bu yapı daha hızlı bir değiştirme ve geliştirme süreci sunar, dolayısıyla bu da verimliliğin arttığı anlamına gelir. Kısa zamanda yüksek verimlilik de maddi kazanç sağlar.

Bunun yanında front-end yapısı için ise firebase i destekleyen **REACT NATİVE** kullanıldı. Kendine has bir dil formatı (JSX) olan React Native bizlere tek bir dil üzerinden kodlama yapabilme ve geliştirilen uygulamanın birçok platformda çalışma olanağını sunuyor. Bize bu desteği sağlayan React Native, cihaz ile arayüz arasında bir köprü görevi görerek geliştirilen mobil uygulamamızın sorunsuzca çalışmasını sağlıyor.

Bir sayfadan başka bir sayfaya geçişimizi sağlamak için **geçiş yığınları (stack navigatör)** modülü kullanırız. React Native ile uygulama geliştirirken olmazsa olmazlarımızdan bir tanesidir. Çünkü uygulamamızda birden fazla sayfa olmaktadır.

React Native' e bağlı olarak **Hooks** yapısı kullanıldı. Hooks; uygulama durumunu yönetmek için açık kaynaklı bir JavaScript kütüphanesidir. Hooks sayesinde üst seviye bileşenden alt seviye bileşene state aktarımına gerek kalmaz. Bu sayede uygulamamızın front-end kısmını tamamlamış olduk.

Projemizi ana hatlarıyla kısaca özetleyerek mimari tasarımımızı sunmuş olduk. Böylece uygulamamız için kullanılan platformları gerekçeleriyle açıkladık.