**AR Project - Kurulum ve Çalıştırma Kılavuzu**

**1. Giriş**

Bu belge, **AR Project** adlı artırılmış gerçeklik (AR) tabanlı mobil uygulamanın nasıl kurulacağı ve çalıştırılacağı konusunda ayrıntılı rehberlik sağlamaktadır. Artırılmış gerçeklik teknolojisi, gerçek dünya ile sanal nesnelerin birleştirilerek kullanıcıya interaktif bir deneyim sunmasını amaçlayan modern bir teknolojidir. **AR Project**, bu teknolojiyi kullanarak mobil cihazlar aracılığıyla kullanıcıya etkileşimli ve zengin bir deneyim sunmayı hedeflemektedir.

Bu rehber, uygulamanın **kurulumu, temel yapılandırması, bağımlılıkların yüklenmesi, çalıştırma süreci, test edilmesi ve olası hata giderme adımları** gibi geniş bir içeriği kapsamaktadır. Aynı zamanda, uygulamanın çalıştırılabilmesi için **gerekli olan tüm yazılım ve araçlar**, bunların **nasıl yüklenmesi gerektiği** ve **karşılaşılabilecek olası sorunlara çözüm önerileri** ayrıntılı olarak ele alınmaktadır.

Uygulama **React Native** çerçevesi kullanılarak geliştirilmiştir ve **Expo** platformu üzerinden çalıştırılmaktadır. Expo, React Native projelerinin hızlı bir şekilde geliştirilmesini, test edilmesini ve dağıtılmasını kolaylaştıran bir araçtır. Bu sayede, uygulama hem **Android** hem de **iOS** platformlarında **aynı kod tabanı** ile çalıştırılabilmektedir. Expo’nun sağladığı avantajlar arasında **hızlı geliştirme döngüsü, canlı yeniden yükleme (hot reload), emülatör veya fiziksel cihaz desteği ve bağımsız çalışabilen bir geliştirme ortamı** bulunmaktadır.

Bu belgedeki adımlar **yeni başlayanlar için bile anlaşılır olacak şekilde** hazırlanmıştır. Proje ile ilgili herhangi bir sorununuz olduğunda, rehberin ilgili bölümlerine başvurarak hızlı çözümler üretebilirsiniz. Kurulumdan hata ayıklamaya kadar tüm adımların detaylı olarak ele alındığı bu rehber, geliştiricilere **AR Project uygulamasını verimli bir şekilde başlatma, geliştirme ve test etme imkanı** sunacaktır.

**2. Gereksinimler**

Uygulamayı başarılı bir şekilde çalıştırmak için aşağıdaki gereksinimlerin eksiksiz bir şekilde karşılandığından emin olun. Eksik veya hatalı bir kurulum, geliştirme sürecinde hatalarla karşılaşmanıza neden olabilir.

* **Node.js (LTS sürümü)**:  
  Uygulamanın çalıştırılması ve bağımlılıkların yönetilmesi için **Node.js** gereklidir. En güncel LTS (Long-Term Support) sürümünü [**Node.js resmi web sitesinden**](https://nodejs.org/) indirerek yükleyin. Kurulum tamamlandıktan sonra terminal veya komut satırında aşağıdaki komutları çalıştırarak kurulumun doğruluğunu kontrol edebilirsiniz:

node -v

npm -v

Bu komutlar sırasıyla Node.js ve npm (Node Package Manager) sürümlerini gösterecektir. Eğer bir hata alırsanız, yüklemeyi tekrar kontrol edin.

* **Expo CLI**:  
  Expo platformu, React Native uygulamalarını kolayca geliştirmek ve test etmek için kullanılır. Expo CLI'yi sisteminize yüklemek için aşağıdaki komutu çalıştırın:

npm install -g expo-cli

Bu yükleme işlemi tamamlandıktan sonra, Expo'nun düzgün çalıştığını doğrulamak için şu komutu kullanabilirsiniz:

expo --version

Eğer sürüm numarası görünmüyorsa veya bir hata mesajı alıyorsanız, yüklemenin doğru şekilde tamamlanmadığını kontrol edin.

* **Git (Opsiyonel - Proje Git üzerinden çekilecekse)**:  
  Projenin GitHub veya başka bir sürüm kontrol sisteminden indirilmesi gerekiyorsa, **Git** kurulu olmalıdır. Git'i [resmi web sitesinden](https://git-scm.com/) indirerek yükleyin.  
  Kurulumun başarılı olup olmadığını anlamak için terminal veya komut satırında şu komutu çalıştırabilirsiniz:

git --version

Eğer bir sürüm numarası dönüyorsa, Git düzgün çalışıyor demektir.

* **Mobil Cihazda Expo Go Uygulaması**:  
  Expo Go, Expo projelerini test etmek için kullanılan bir mobil uygulamadır. Bu uygulama hem **Android** hem de **iOS** cihazlarda çalışır ve proje geliştirme sürecinde gerekli bir bileşendir.
  + **Android kullanıcıları**, Google Play Store üzerinden Expo Go uygulamasını indirebilir.
  + **iOS kullanıcıları**, [App Store](https://apps.apple.com/us/app/expo-go/id982107779) üzerinden Expo Go uygulamasını yükleyebilir.

Mobil cihazınızın **aynı Wi-Fi ağına** bağlı olması gereklidir. Expo üzerinden oluşturulan QR kodu taratılarak uygulama test edilebilir.

* **Android Studio ve iOS Simulator (Emülatör Kullanımı İçin Önerilir)**:  
  Eğer uygulamayı bir fiziksel cihazda değil de bilgisayar üzerindeki bir **emülatör (simülatör)** kullanarak çalıştırmak istiyorsanız, aşağıdaki yazılımları yüklemeniz önerilir:
  + **Android Studio**: React Native projelerini test etmek için **Android Emulator** kullanabilirsiniz. [Android Studio’nun resmi web sitesinden](https://developer.android.com/studio) en güncel sürümü indirip yükleyin.
    - Yükleme sonrası **AVD Manager** üzerinden sanal cihaz oluşturarak uygulamayı çalıştırabilirsiniz.
    - adb devices komutuyla cihazın başarılı şekilde tanındığını doğrulayabilirsiniz.
  + **iOS Simulator**: iOS cihazlarda test yapmak için **macOS işletim sistemine** ve **Xcode** uygulamasına ihtiyacınız vardır.
    - **Xcode’u** Mac App Store’dan indirip kurduktan sonra, **Xcode Developer Tools** bölümünden iOS Simulator’ü açabilirsiniz.
    - xcrun simctl list devices komutuyla mevcut sanal cihazları listeleyebilirsiniz.

**3. Kurulum**

Kurulum süreci, uygulamanın çalıştırılabilmesi için gerekli olan **kod dosyalarının edinilmesi, bağımlılıkların yüklenmesi ve gerekli yapılandırmaların tamamlanması** adımlarını içermektedir. Aşağıda adım adım kurulum süreci açıklanmıştır. Bu adımları eksiksiz bir şekilde takip ederek **AR Project** uygulamasını sorunsuz bir şekilde çalıştırabilirsiniz.

**Adım 1: Proje Dosyalarının İndirilmesi ve Yerel Ortama Kopyalanması**

Öncelikle proje dosyalarını edinmeniz gerekmektedir. Proje, GitHub gibi bir sürüm kontrol sisteminde barındırılıyorsa, **Git** kullanarak projeyi yerel bilgisayarınıza klonlayabilirsiniz.

Bunun için terminal veya komut satırını açarak aşağıdaki komutu çalıştırın:

git clone <proje-repo-url>

Bu komut, projenin tüm dosyalarını belirttiğiniz **proje-repo-url** adresinden bilgisayarınıza indirecektir.

Alternatif olarak, eğer Git kullanmıyorsanız:

* GitHub web sitesine giderek projeyi **ZIP dosyası olarak indirin** ve bilgisayarınızda uygun bir dizine çıkartın.

İndirme işlemi tamamlandıktan sonra, proje dizinine geçiş yapmak için aşağıdaki komutu kullanın:

cd AR-project-main

Bu komut, terminali veya komut satırını **AR Project** uygulamasının bulunduğu klasöre yönlendirecektir.

**Adım 2: Proje Bağımlılıklarının Yüklenmesi**

Projeyi başarılı bir şekilde çalıştırabilmek için, Node.js paket yöneticisi **npm (Node Package Manager)** kullanılarak bağımlılıkların yüklenmesi gerekmektedir. Proje dizinindeyken aşağıdaki komutu çalıştırın:

npm install

Bu komut, package.json dosyasında tanımlanmış olan tüm bağımlılıkları yükleyerek projenin eksiksiz çalışmasını sağlar. Bu işlem internet hızınıza bağlı olarak birkaç dakika sürebilir. İşlem sırasında hata alırsanız:

* **Node.js ve npm sürümlerinizin güncel olup olmadığını** kontrol edin.
* Eğer npm install komutu eksik bağımlılıklar nedeniyle hata veriyorsa, aşağıdaki komutları sırasıyla deneyebilirsiniz:

npm cache clean --force

npm install

* Alternatif olarak, bağımlılıkları sıfırdan yüklemek için:

rm -rf node\_modules package-lock.json

npm install

**Adım 3: Expo ile Gerekli Bileşenlerin Yüklenmesi**

Expo, React Native projelerinde çeşitli paketleri yönetmek için kullanılan bir çerçevedir. Bağımlılıkların yüklenmesinin ardından **Expo'nun gerektirdiği ek bileşenlerin yüklenmesi** için aşağıdaki komutu çalıştırın:

expo install

Bu komut, Expo tarafından önerilen ve proje için kritik olan paketleri yükler. Expo, bağımlılıkları doğrudan proje ile uyumlu olan en uygun sürümde indirir.

Eğer bu adımda expo install komutunda bir hata alırsanız:

* **Expo CLI'nin güncellenmiş olduğundan emin olun**. Güncellemek için aşağıdaki komutu çalıştırabilirsiniz:

npm install -g expo-cli

* **Bağımlılıkların doğru şekilde yüklenmesini sağlamak için** aşağıdaki komutları sırasıyla çalıştırabilirsiniz:

expo doctor

expo upgrade

Kurulum işlemi tamamlandıktan sonra uygulamanın çalıştırılması için bir sonraki adıma geçebilirsiniz.

**4. Uygulamayı Çalıştırma**

Uygulamanın başarıyla çalıştırılabilmesi için, Expo geliştirme sunucusunun başlatılması ve proje dosyalarının uygun bir ortamda çalıştırılması gerekmektedir. Aşağıdaki adımları eksiksiz bir şekilde takip ederek **AR Project** uygulamasını hem fiziksel cihazda hem de emülatör ortamında çalıştırabilirsiniz.

**Adım 1: Expo Geliştirme Sunucusunu Başlatma**

Expo geliştirme sunucusunu çalıştırmak için terminal veya komut satırını açarak aşağıdaki komutu girin:

npx expo start

Bu komut, proje kök dizininde bulunan app.json ve package.json dosyalarını okuyarak **Expo geliştirme ortamını** başlatacaktır. Komut çalıştırıldıktan sonra terminalde **Expo geliştirici arayüzü** açılacaktır.

npm start

**Expo geliştirme sunucusunun avantajları:**

* Otomatik olarak yerel geliştirme ortamını başlatır.
* **QR kod** ile mobil cihazlardan kolayca test edilmesini sağlar.
* **Emülatör ve fiziksel cihaz desteği** sunar.
* **Hata ayıklama araçları** ile uygulama geliştirme sürecini kolaylaştırır.

**Adım 2: Mobil Cihaz Üzerinde Çalıştırma**

Expo geliştirme sunucusu başlatıldıktan sonra, terminalde **Expo geliştirici arayüzü** açılacaktır. Burada **QR kodu tarayarak uygulamayı mobil cihazınızda çalıştırabilirsiniz.**

**Mobil cihazda çalıştırmak için:**

1. Mobil cihazınıza **Expo Go** uygulamasını yüklediğinizden emin olun.
   * **Android kullanıcıları**: Google Play Store
   * **iOS kullanıcıları**: [App Store](https://apps.apple.com/us/app/expo-go/id982107779)
2. Terminalde veya Expo arayüzünde görüntülenen **QR kodunu tarayın**.
3. **Expo Go uygulaması** ile projeyi açarak test etmeye başlayın.

Eğer QR kodu kullanamıyorsanız, **Expo projelerinin URL’sini manuel olarak girerek** çalıştırabilirsiniz. Terminalde görüntülenen exp:// ile başlayan bağlantıyı kopyalayarak Expo Go uygulamasına yapıştırabilirsiniz.

**Adım 3: Emülatör veya Simülatör Üzerinde Çalıştırma**

Eğer uygulamayı bir **fiziksel cihaz** yerine **bilgisayardaki sanal cihazlarda** test etmek istiyorsanız, aşağıdaki adımları takip edebilirsiniz.

**Android Emulator Kullanımı (Android Studio ile Çalıştırma)**

1. **Android Studio**’nun yüklü olduğundan emin olun.
2. Android Studio’yu açarak **Virtual Device Manager (AVD Manager)** üzerinden bir sanal cihaz oluşturun.
3. Terminalde expo start komutunu çalıştırdıktan sonra **Expo geliştirici arayüzünde "Run on Android device/emulator"** seçeneğine tıklayın.
4. Uygulamanız **Android Emulator** üzerinde çalışmaya başlayacaktır.

Alternatif olarak, terminalden doğrudan şu komutu kullanarak emülatörde çalıştırabilirsiniz:

npx expo run:android

**iOS Simulator Kullanımı (macOS ve Xcode Gerektirir)**

1. **Xcode ve Xcode Command Line Tools** yüklü olmalıdır.
2. Terminalde expo start komutunu çalıştırdıktan sonra Expo geliştirici arayüzünde **"Run on iOS simulator"** seçeneğini seçin.
3. Eğer expo start komutuyla iOS Simulator çalışmazsa, aşağıdaki komutu kullanabilirsiniz:

npx expo run:ios

1. Uygulamanız **iOS Simulator** üzerinde çalışacaktır.

**Not:**

* iOS simülatörünü yalnızca **macOS işletim sisteminde** kullanabilirsiniz.
* Apple Developer hesabınızın olması bazı durumlarda gereklidir.

**Adım 4: Uygulamanın Canlı Güncellenmesini Sağlama**

Expo’nun sunduğu en büyük avantajlardan biri **canlı yeniden yükleme (Live Reload) ve hızlı güncelleme (Hot Reload)** özelliğidir. Kodda yapılan değişikliklerin anında yansımasını sağlamak için aşağıdaki komutu kullanabilirsiniz:

expo start --tunnel

Bu komut, **tünel bağlantısı** oluşturarak uygulamanın **internet üzerinden erişilebilir** olmasını sağlar.

* **--lan seçeneği**: Aynı ağdaki cihazlarda çalıştırmak için kullanılır.
* **--localhost seçeneği**: Sadece kendi bilgisayarınızda çalıştırmak için kullanılır.
* **--tunnel seçeneği**: Farklı ağlarda veya dış bağlantılar için kullanılır.

Alternatif olarak, uygulamanın **geliştirme sürecinde anlık olarak güncellenmesini** sağlamak için aşağıdaki adımları da takip edebilirsiniz:

* Expo geliştirici arayüzünde **"Enable Fast Refresh"** seçeneğini etkinleştirin.
* Kod değiştirildiğinde otomatik olarak yenilenmesini sağlamak için:

expo r -c

Bu komut, Expo önbelleğini temizleyerek **tüm güncellemelerin düzgün çalışmasını** sağlayacaktır.

**5. Yapılandırma**

**AR Project** uygulamasının doğru şekilde çalışabilmesi için çeşitli yapılandırma dosyaları bulunmaktadır. Bu dosyalar, uygulamanın temel ayarlarını yönetmek, bağımlılıkları düzenlemek ve geliştirme sürecini optimize etmek amacıyla kullanılır. Aşağıda **app.json**, **metro.config.js** ve **babel.config.js** dosyalarının detaylı açıklamaları verilmiştir.

**1. app.json Dosyası**

Bu dosya, **Expo ve React Native** projelerinde uygulama adı, simgeleri (ikonlar), başlangıç ekranı (splash screen) ve genel yapılandırmaları belirlemek için kullanılır. **app.json** dosyası aşağıdaki gibi önemli parametreleri içerir:

**Temel Yapılandırmalar:**

* **"name"**: Uygulamanın iç sistemdeki adı.
* **"slug"**: Expo tarafından kullanılan proje kimliği.
* **"version"**: Uygulamanın sürüm numarası.
* **"orientation"**: Uygulamanın **"portrait"** (dikey) veya **"landscape"** (yatay) modda çalışmasını belirler.
* **"platforms"**: Hangi işletim sistemlerinde çalışacağını belirtir (**"android", "ios", "web"**).

**Görsel Ayarlar:**

* **"icon"**: Uygulamanın ana simgesi, **assets/icon.png** gibi bir dosya ile tanımlanır.
* **"splash"**: Uygulamanın açılış ekranı (Splash Screen) görüntüsünü ayarlar. Örnek:

"splash": {

"image": "./assets/splash.png",

"resizeMode": "contain",

"backgroundColor": "#ffffff"

}

* **"ios" ve "android" ayarları**: Her platform için ayrı ayarlar belirlenebilir.

"ios": {

"supportsTablet": true

},

"android": {

"adaptiveIcon": {

"foregroundImage": "./assets/adaptive-icon.png",

"backgroundColor": "#ffffff"

}

}

Eğer uygulamanın başlangıç ekranı düzgün çalışmıyorsa, **assets klasöründeki görsellerin doğru adlandırıldığını ve uygun formatta (PNG) olduğunu kontrol edin.**

**2. metro.config.js Dosyası**

Bu dosya, **Metro Bundler** yapılandırmasını yönetir. **Metro**, React Native projeleri için bir JavaScript paketi yükleyicidir ve kodları optimize etmek için kullanılır.

**Metro.config.js Dosyasının Önemi**

* **Proje bağımlılıklarının nasıl yükleneceğini tanımlar.**
* **Özel varlıkların (resimler, yazı tipleri, animasyonlar) nasıl ele alınacağını belirler.**
* **Hata ayıklama sürecini hızlandırır ve performans iyileştirmeleri sunar.**

Örnek bir **metro.config.js** dosyası:

const { getDefaultConfig } = require('expo/metro-config');

module.exports = (async () => {

const config = await getDefaultConfig(\_\_dirname);

config.resolver.assetExts.push('cjs'); // Ek dosya formatlarını destekleme

return config;

})();

Bu yapılandırma, **CJS (CommonJS) dosyalarını** desteklemek için kullanılabilir.

**Metro.config.js ile Yapılabilecek Değişiklikler:**

* **Yeni dosya türleri eklemek** (Örneğin .ttf, .svg, .cjs gibi).
* **Hata ayıklama ayarlarını değiştirmek**.
* **React Native projelerinde hızlandırılmış derleme sağlamak**.

Eğer proje sırasında **bundling** (paketleme) sorunlarıyla karşılaşırsanız, aşağıdaki komutu çalıştırarak Metro Bundler'ı sıfırlayabilirsiniz:

expo r -c

Bu işlem, önbelleği temizleyerek **proje derleme hatalarını** düzeltebilir.

**3. babel.config.js Dosyası**

Bu dosya, **JavaScript derleyicisi olan Babel’in yapılandırmalarını** içerir. **Babel**, modern JavaScript kodlarını **daha eski tarayıcılar ve cihazlar** için dönüştürerek uyumluluk sağlar.

**Babel.config.js'in Temel Kullanımı:**

* **ES6 ve üstü JavaScript kodlarını eski sürümlere çevirmek için kullanılır.**
* **React Native projelerinde kullanılacak modüllerin ve özelliklerin ayarlanmasını sağlar.**
* **Kodların performansını artırmaya yönelik optimizasyonlar içerir.**

Örnek bir **babel.config.js** dosyası:

module.exports = function(api) {

api.cache(true);

return {

presets: ['babel-preset-expo'],

plugins: [

['react-native-reanimated/plugin'], // Performansı artıran bir eklenti

],

};

};

**Babel.config.js İçin Önemli Ayarlar:**

* **"presets"**: Expo için ön tanımlı Babel ayarlarını içerir.
* **"plugins"**: Performans artırıcı veya özel işlemleri destekleyen eklentileri yükler.

Eğer derleme hatası alıyorsanız, aşağıdaki komutları sırasıyla çalıştırarak sorunu çözebilirsiniz:

npm cache clean --force

npm install

expo r -c

**Genel Yapılandırma Özet**

| **Dosya Adı** | **Açıklama** |
| --- | --- |
| **app.json** | Uygulama adı, simgeleri, splash ekranı ve genel ayarları içerir. |
| **metro.config.js** | Proje bağımlılıklarını yönetir ve Metro Bundler ayarlarını belirler. |
| **babel.config.js** | JavaScript kodlarının derlenmesi için Babel yapılandırmasını içerir. |

Bu yapılandırma dosyaları, **uygulamanın kararlı ve verimli bir şekilde çalışmasını sağlamak için düzenli olarak kontrol edilmelidir.** Özellikle **React Native ve Expo sürümleri güncellendiğinde, yapılandırma dosyalarının da uyumlu olup olmadığı** test edilmelidir.

**6. Hata Giderme**

Uygulama geliştirme sürecinde, çeşitli hatalarla karşılaşabilirsiniz. Aşağıda, yaygın hatalar ve bunların nasıl çözülebileceği ayrıntılı bir şekilde açıklanmaktadır. Hataların büyük çoğunluğu **bağımlılık sorunları, bağlantı problemleri ve emülatör yapılandırmaları** ile ilgilidir.

**1. npm install Komutuyla İlgili Hatalar**

Proje bağımlılıklarını yüklemek için kullanılan npm install komutu çalıştırıldıktan sonra hata alıyorsanız, aşağıdaki adımları uygulayın:

* **Node.js ve npm sürümlerinizin güncel olup olmadığını kontrol edin**:

node -v

npm -v

Eğer Node.js veya npm eski bir sürümse, bunları güncelleyin. Güncellemek için:

npm install -g npm

Alternatif olarak, [Node.js resmi web sitesinden](https://nodejs.org/) en güncel sürümü indirerek yeniden yükleyebilirsiniz.

* Eğer bağımlılıklar eksik yükleniyorsa veya package-lock.json hataları alıyorsanız, aşağıdaki komutları çalıştırarak bağımlılıkları sıfırlayın:

rm -rf node\_modules package-lock.json

npm install

**2. Expo CLI ile İlgili Hatalar**

Expo ile ilgili bir hata alıyorsanız, öncelikle Expo CLI’nin güncel olup olmadığını kontrol edin ve gerekirse aşağıdaki komutu kullanarak güncelleyin:

npm install -g expo-cli

Expo CLI’nin düzgün çalışıp çalışmadığını test etmek için aşağıdaki komutu çalıştırabilirsiniz:

expo diagnostics

Bu komut, Expo’nun mevcut sistem yapılandırmasını ve olası hataları gösterir.

**3. expo start Bağlantı Sorunları**

Eğer expo start komutunu çalıştırdıktan sonra **bağlantı sorunları yaşıyorsanız**, aşağıdaki adımları deneyebilirsiniz:

* **Bağlantıyı sıfırlayın**

expo start --clear

Bu komut, önbelleği temizleyerek bağlantı ile ilgili hataları düzeltebilir.

* **Cihazın veya emülatörün aynı ağda olduğundan emin olun.**
* **Bağlantı ayarlarını değiştirerek test edin:**

expo start --lan

veya

expo start --tunnel

**LAN modu**, aynı ağdaki cihazlar için en iyi performansı sağlar. Ancak, farklı ağlarda test yapmak için **Tunnel modu** önerilir.

**4. Android Emülatör Sorunları**

Eğer Android emülatörü başlatılamıyorsa, aşağıdaki adımları kontrol edin:

* **ADB Bağlantısını Kontrol Edin**

adb devices

Eğer bağlı bir cihaz veya emülatör görünmüyorsa, Android Studio üzerinden **AVD Manager**'ı açarak sanal cihazı başlatın.

* **ADB Servisini Yeniden Başlatın**

adb kill-server

adb start-server

* **Hala çalışmıyorsa, Expo için doğrudan emülatör başlatın**

npx expo run:android

**5. iOS Emülatör Sorunları**

iOS emülatörünü çalıştırırken sorun yaşıyorsanız:

* **Xcode’un güncel olduğundan emin olun** ve komut satırında şu komutları çalıştırarak mevcut cihazları listeleyin:

xcrun simctl list devices

* **iOS simülatörünü manuel olarak başlatın**:

open -a Simulator

* **Expo'yu iOS simülatöründe çalıştırmak için aşağıdaki komutu kullanabilirsiniz:**

npx expo run:ios

**7. Ekstra Özellikler**

Expo, geliştiricilere **hızlı kod değişiklikleri yapma, ağ bağlantılarını özelleştirme ve canlı güncellemeler ile test etme** gibi ek özellikler sunar.

**1. Hot Reload (Canlı Yeniden Yükleme) Özelliği**

**Hot Reload**, geliştirme sürecinde kod değişikliklerinin anında uygulamaya yansıtılmasını sağlar. Bu sayede, her değişiklikten sonra uygulamayı yeniden başlatmanıza gerek kalmaz.

**Hot Reload’u etkinleştirmek için:**

* **Geliştirici menüsünü açın**
* **"Enable Fast Refresh"** seçeneğini etkinleştirin

Alternatif olarak, terminalde şu komutu çalıştırarak **önbelleği temizleyebilir** ve güncellemeleri alabilirsiniz:

expo r -c

**2. Bağlantı Seçenekleri ve Kullanım Alanları**

Expo projelerini farklı bağlantı türleri üzerinden çalıştırabilirsiniz:

* **Yerel Makinede Çalıştırma (Offline Kullanım için)**

expo start --localhost

**Kullanım Alanı:** Uygulamanın yalnızca geliştirici bilgisayarında test edilmesini sağlar.

* **Aynı Ağ İçindeki Cihazlara Bağlanma**

expo start --lan

**Kullanım Alanı:** Telefon ve bilgisayar **aynı Wi-Fi ağına bağlıysa** hızlı bağlantı sunar.

* **Farklı Ağlar veya Dış Bağlantılar için (Uzaktan Erişim)**

expo start --tunnel

**Kullanım Alanı:** Cihazınız farklı bir ağda veya mobil veri üzerinden bağlanıyorsa önerilir.

**8. Sonuç**

Bu rehber, **AR Project** uygulamasının nasıl kurulacağı, yapılandırılacağı ve çalıştırılacağı hakkında **detaylı bilgiler sunmaktadır**. Adım adım kurulum ve hata giderme süreçleri açıklanarak, olası sorunların nasıl çözülebileceği detaylandırılmıştır.

Uygulamanın başarılı bir şekilde geliştirilmesi ve test edilmesi için şu kaynakları kullanabilirsiniz:

* **Expo’nun resmi dokümantasyonu**: https://docs.expo.dev/
* **React Native dokümantasyonu**: https://reactnative.dev/docs/getting-started
* **Geliştirici forumları ve Stack Overflow**, olası hatalar ve çözümleri hakkında geniş bir bilgi kaynağı sunmaktadır.

Daha fazla bilgi için proje dizinindeki **README.md** dosyasını inceleyebilir veya **topluluk forumları** üzerinden destek alabilirsiniz. Eğer hâlâ çözülmemiş bir hata varsa, terminalde expo diagnostics komutunu çalıştırarak ayrıntılı hata raporunu inceleyebilir ve uygun çözümü belirleyebilirsiniz.