## BÖLÜM 12

Firebase'i Flutter ile Kullanma

Bir girişimci/mentor olarak bana iş uygulamaları için bir sürü harika fikir sunuldu. İstisnasız her biri veri depolayan bir sunucu içeriyordu. Dolayısıyla, Flutter'ın gerçek dünyadaki kullanımını önemsiyorsak, üretime hazır bir sunucuyla konuşmayı göz ardı etmek suç olacaktır.

Amazon AWS, Microsoft Azure, IBM Cloud, Oracle Cloud Infrastructure, Alibaba Cloud ve daha pek çok sunucu çözümü mevcut. Hepsi harika. Firebase'e odaklanmayı seçtik, çünkü diğerlerinden çok daha iyi olduğu için değil

1. Firebase ve Flutter'ın her ikisi de Google ürünüdür, bu nedenle birkaç sinerji vardır.
2. Firebase en az diğer seçenekler kadar iyi ve bazı

yönlerden onlardan daha iyi.

1. Firebase'in kurulumu (oldukça) kolaydır ve düşük hacimler için ücretsizdir - öğrenme ve test için mükemmeldir.

© Rap Payne 2019

R. Payne, *Flutter ile Uygulama Geliştirmeye Başlarken*, [https://doi.org/10.1007/978-1-4842-5181-](#_bookmark0) [2\_12](#_bookmark0) 255

1. Firebase son zamanlarda geliştirici topluluğunda büyük ilgi görüyor. İşe alım yöneticilerinin ve işe alım uzmanlarının özgeçmişinizde görmek istediği bir teknoloji haline geliyor.
2. Birini seçmek zorundaydık, neden Firebase olmasın? ¯\\_(ツ)\_/¯

Firebase'i Flutter ile kullanmak için bir sunucu olarak uygulayacaksak, bunları anladığımızdan emin olmamız gerekir:

* + Çok yüksek seviyede Firebase
  + Firebase projesi nasıl kurulur
  + Nihai iOS ve Android uygulamalarımızla nasıl çalışır?
  + Yetkilendirme ile Flutter uygulamamıza entegre etme
  + Firestore verilerini Flutter uygulamamıza okuma
  + Flutter'da veri sorgulama
  + Flutter'da veri değiştirme
  + Flutter'da veri silme

Bu bölüm için oyun planımız bu olacak. Firebase'in kendisine bir giriş ile başlayalım.

# Firebase ile tanışın

Google Firebase, sunucu tarafı hizmetleri ve araçlarından oluşan bir settir. Firebase kullanıyorsanız, kendi sunucunuzu satın almanıza veya kiralamanıza gerek yoktur. Güvenlik yamalarını uygulamak veya yazılımı güncellemek yok. Yedekleme düzenlemek yok. Güvenlik duvarlarının yapılandırılması yok. Saldırı tespit sistemleri yok. Kötü amaçlı yazılımdan koruma tanımı bakımı yok.

İnternete bağlanmak için bir İSS'ye ödeme yapmak yok.

Temel olarak, Google'a Google ile yapacağınız her şeyi halletmesi için .

kendi sunucunuz. Elbette bir sunucu var, ancak sizin ve benim işletim sistemine erişimimiz yok, bu nedenle ne sorumluluğumuz ne de bakımını yapma yeteneğimiz var. Bu ne kadar özgürleştirici?

Firebase ile yapamayacağınız şeylerin listesi küçüktür ve yaygın olarak ihtiyaç duyulmaz. Ancak yapabileceklerinizin listesi geniş ve çok yaygındır. Firebase bir düzineden fazla araçtan oluşur.[1](#_bookmark1) Bu üçüne bir göz atalım:

* Cloud Firestore - Veri okumak ve yazmak için API'ye sahip bir veritabanı
* Bulut İşlevleri - Bir API çağrısı tarafından başlatılan mantık
* Kimlik Doğrulama - Kullanıcıların sosyal hesaplarını veya kullanıcı adı/şifre kombinasyonunu kullanarak uygulamanıza güvenli bir şekilde giriş yapmalarını sağlamak için tek oturum açma

Her birini kısaca tartışalım ve ardından geçen bölümdeki örnek Flutter

uygulamasını Firestore'dan okuma ve yazma için genişletelim.

## Bulut Firestore

Tüm uygulamaların okuyabileceği ve yazabileceği bir veritabanı istiyoruz. Firestore, bu verilere erişmek için depolama ve araçlara sahip bir NoSQL veritabanı sağlar. NoSQL olduğu için oldukça esnektir, belki de rahat ettiğinizden daha esnektir. MongoDB, Cassandra veya CouchDB ile çalıştıysanız, burada neden bahsettiğimizi biliyorsunuzdur. Firestore'un en büyük farkı, veritabanının kendisinin ve arka uç motorunun sizin ve benim yerime Google tarafından sürdürülüyor olmasıdır. Tek yapmamız gereken verilere erişmek.

Firestore, veriler için API uç noktaları sunar. Uygulamanız kendisini

Firestore'a tanıttıktan sonra, bu uç noktalardan veri okuyabilir ve yazabilir. Geçen bölümde tartıştığımız RESTful arayüzüne benziyor, değil mi?

1 Hepsi hakkında buradan bilgi [edinin: https://firebase.google.com/products](https://firebase.google.com/products)

Firestore, REST'in bazı özelliklerine sahip bir arayüzü destekliyor, ancak RESTful olarak kategorize edemeyeceğim kadar farklı. Bunun yerine, uygulamamızın kimliğini doğrulama ve özel iletişim kurma gibi ağır işleri üstlenen bir Dart kütüphanesi kullanıyoruz. Bu kütüphanedeki Firestore.get('people') veya Firestore.set('categories') gibi yöntemleri çağıracağız. Bu, bir kez ayarladığımızda çok daha akıcı hale geliyor. (Ancak kurulum süper basit değil.)

## Bulut İşlevleri

Uygulamanız neredeyse kesinlikle veri işlemeyle ilgilidir. Bu algoritmalardan bazıları çok ama çok karmaşık olabilir. Ancak gerçek şu ki, kullanıcılarınızın cihazları muhtemelen bu işleri halledebilecek kapasiteden daha fazlasına sahiptir.

"Peki neden her şeyi uygulamamda işlemiyorum? " diye soruyorsunuz. Çünkü eğer cihaz üzerindeyse, algoritmalar herhangi biri tarafından tersine mühendislik

Flutter uygulamanızı indiren saldırgan. Tüm gizli iş süreçleri açığa çıkar ve mantık . API anahtarlarınızdan herhangi biri cihazınızda saklanır ve okunabilir. Bu yüzden cihaza herhangi bir hassas veri veya işlem koymamayı tercih ederiz. Bunları erişilemeyecekleri bir yere, sunucuya koyalım.

Peki ya iki veya daha fazla uygulama arasında paylaşılması gereken işlemler veya veriler? Bunlar için bir sunucuya ihtiyacınız var.

Bulut İşlevlerini aşağıdaki gibi şeyler için istersiniz

* Üçüncü taraf bir API tüketme
* Elektronik tablolar gibi sunucu tarafı dosyalarının işlenmesi
* Büyük veri setlerini ayıklama, işleme, dönüştürme ve yükleme
* Sohbet robotu veya sohbet uygulaması çalıştırma
* Yüz algılama veya metin tanıma ve çıkarma gibi görüntü analizi
  + Saldırgan görüntüleri bulanıklaştırma gibi büyük görüntü işleme
  + Niyet tespiti gibi metin analizi
  + Makine öğrenimi ve yapay zeka
  + Walmart veya Amazon gibi bir e-ticaret mağazasından ürün sipariş etmek

Bulut İşlevleri JavaScript ile yazılır ve Firestore'a bir kayıt eklendiğinde veya Firestore'da güncellendiğinde, bir kullanıcı oturum açtığında veya birisi belirli bir URL'ye Ajax isteği yaptığında Google'ın sunucularındaki bir Node ortamında talep üzerine çalışır.

## Kimlik Doğrulama

Firebase Authentication, uygulamanıza kimlik doğrulama (nispeten) kolaylaştırır. Elbette, uygulamanıza kaba kuvvet kullanarak kullanıcı adları ve parolalar ekleyebilirsiniz, ancak kullanıcı tablolarını ayarlamak, kimlik doğrulama mantığını yazmak, parolaları karma hale getirmek, unutulan parolaları işlemek ve tüm bunlar için endişelenmeniz gerekir. Firebase Authentication ile, Facebook, Github üzerinden kimlik doğrulama ile birlikte tüm bu işlevleri elde edersiniz,

Twitter ve tabii ki Google'ın kendisi. Kullanıcılarınız kendi kullanıcı adı/şifre kombinasyonlarını kullanmayı seçebilir veya hatta mobil cihazlarında SMS mesajı ile iki faktörlü kimlik doğrulama yapabilirler.

# Firebase'in kendisini kurma

Tüm bu özellikler ve daha fazlası Firebase ile kullanılabilir. Firebase'i denemek istiyorsanız, eğlenceli ve ücretsizdir ve harika bir öğrenme deneyimidir. Ayrıca, bize yeni edindiğimiz Ajax bilgimizi canlı bir okuma/yazma ortamında deneme fırsatı verecektir. Önümüzdeki sayfalarda amacımızın bu olmasına izin vereceğiz.

Öncelikle, Google'da bir hesabınızın olması gerekir. Google hesabınız yoksa, Google'a el yazısıyla bir mektup yazın, üzerine bir pul ve "Google Inc., Mountain View, CA 94043["](#_bookmark2) adresine postalayın.

Devam edin ve Google Hesabınızda oturum açın ve [http://](http://firebase.google.com/) adresini ziyaret edin Firebase ile bir uygulama kaydetmek için firebase[.google.com.](http://firebase.google.com/) Yönergeleri

izleyin. Temel hesap için herhangi bir taahhütte bulunmayacak veya para ödemeyeceksiniz.

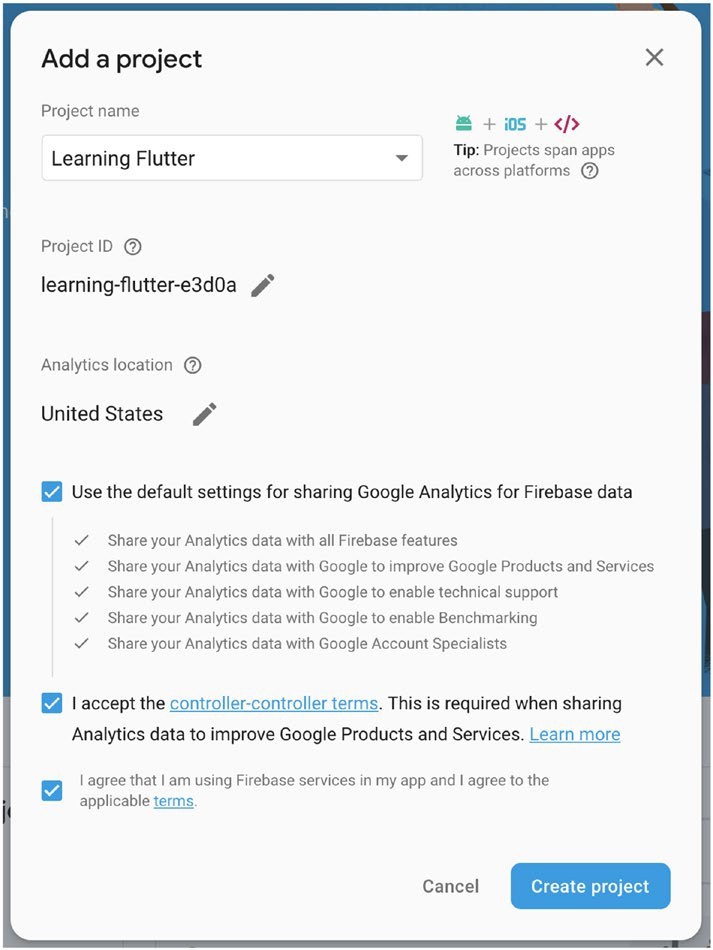
Kariyeriniz boyunca muhtemelen birden fazla projede yer alacaksınız; bazıları öğrenme amaçlı, bazıları yan işleriniz için ve hatta belki bazıları ana işiniz için. Bu nedenle Firebase, kullanıcıların birden fazla projeye sahip olmasına izin verir. Çalışmak için bir tane oluşturacağız.

**Dikkat** Aşağıdaki adımlar yazının yazıldığı tarih itibariyle günceldir, ancak değişebilirler. En güncel adımlar için buraya bir göz atın: [https:](https://firebase.google.com/docs/flutter/setup)//firebase.google.com/docs/flutter/setup.

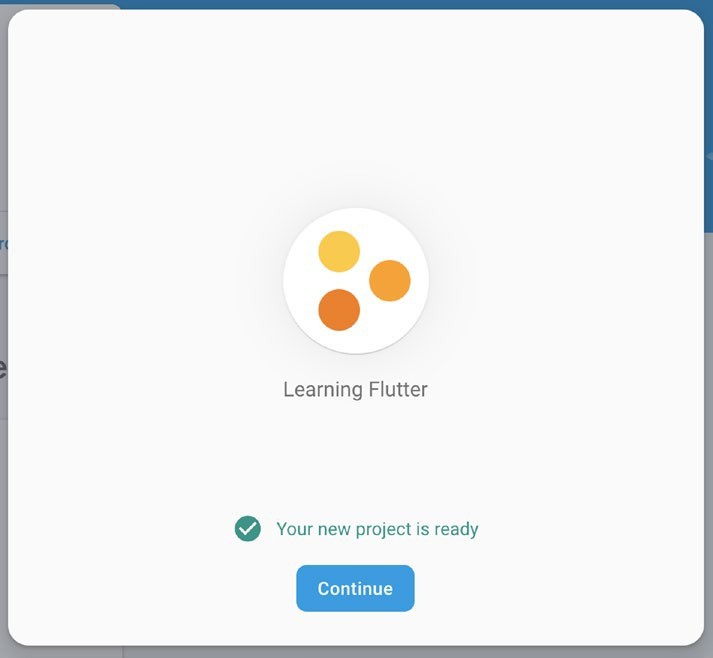
## Firebase projesi oluşturma

Google'da oturum açtıktan sonra console.firebase.google.com adresini ziyaret edin ve sonunda projelerinizin bir listesini içeren konsolunuzu göreceksiniz. Yeni bir proje oluşturmak için düğmeye tıklayın. Projeye "Learning Flutter" gibi bir isim verin (Şekil [12-1](#_bookmark3)). Şekil [12-2](#_bookmark4), projenin kullanıma hazır olduğunu göstermektedir.

2 Şaka yapıyorum. Kimin Google hesabı yok ki?!? Git bir tane aç!



***Şekil 12-1.*** *Yeni bir proje ekleme*



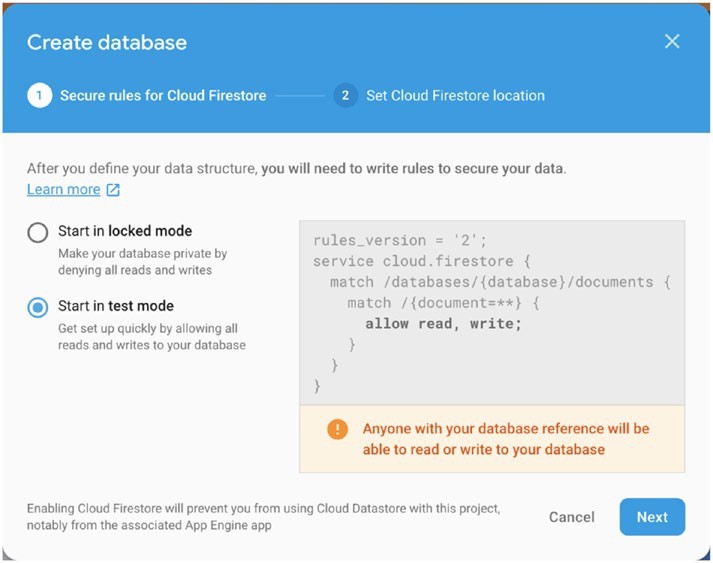
***Şekil 12-2.*** *Projenizin kullanıma hazır olduğuna dair onay*

Şimdi bu projenin bir web arayüzüne ve bir veritabanı arka ucuna sahip olduğunu ve iPhone'lardaki ve Android'lerdeki bir Flutter uygulaması tarafından erişildiğini hayal edin. Bu, birden fazla uygulaması olan bir proje olacaktır.

Birden fazla projede olduğu gibi, Google her projenin birden fazla uygulamaya sahip olmasına izin verir. Ortamların hepsinin farklı talepleri olduğu için her uygulamanın kendi ayarları olacaktır. İşimiz bitmeden önce, iOS ve Android için birer tane kuracağız. Ama önce veritabanımızı ve en az bir koleksiyon (diğer adıyla tablo) oluşturmalıyız.

## Veritabanı oluşturma

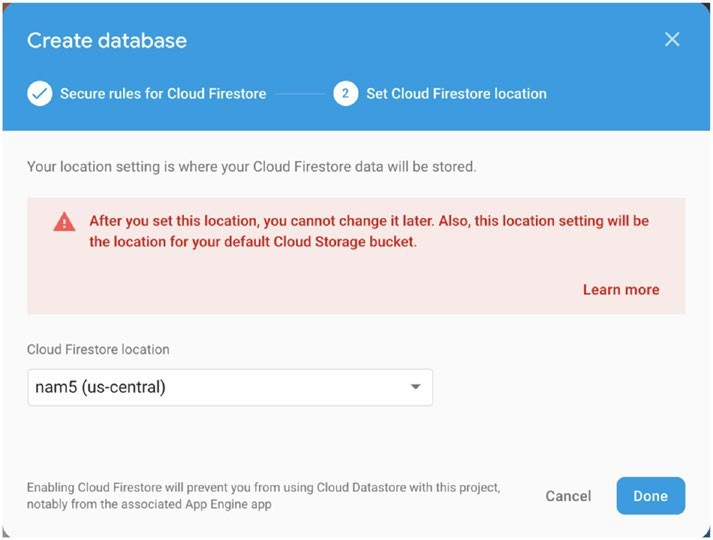
Projenizin kontrol paneline geri dönün. Bir veritabanı ile geliştirme yapmak için bir menü seçeneği göreceksiniz. Devam edin ve yeni bir veritabanı oluşturmayı seçin (Şekil [12-3](#_bookmark5)).



***Şekil 12-3.*** *Cloud Firestore veritabanı oluşturmanın ilk adımı*

Kodumuzun çalıştığını kolayca doğrulayabilmek için test modunda başlatmayı seçin. İstediğiniz zaman ekgerçek dünyadaki bir uygulamaya kurallar eklemek isteyeceksiniz. "İleri" düğmesine basın.

Bu adım, kullanıcılarınızın çoğunun fiziksel olarak nerede bulunacağını sormaktadır (Şekil [12-4](#_bookmark6)). Mevcut konumunuza göre tahminde bulunur ve dürüst olmak gerekirse, herhangi bir konum iyi çalışacaktır. Sadece varsayılanı alın ve "Bitti" düğmesine basın. Bu işlem veritabanını oluşturur.



***Şekil 12-4.*** *Kullanıcılarınızın kısmına en yakın konumu seçin*

**İpucu** Bu, MySQL, SQL Server, Oracle, Informix ve benzerleri gibi geleneksel ilişkisel veritabanlarından farklı olan bir NoSQL veritabanıdır. İlk olarak, terminoloji farklıdır (Tablo [12-1)](#_bookmark7).

***Tablo 12-1.*** *Farklı DB sunucu türlerinde nesnelere nasıl atıfta bulunursunuz*

İ**li**ş**kisel veritabanları NoSQL veritabanları**

Masalar Koleksiyonlar

Kayıtlar/satırlar Belgeler

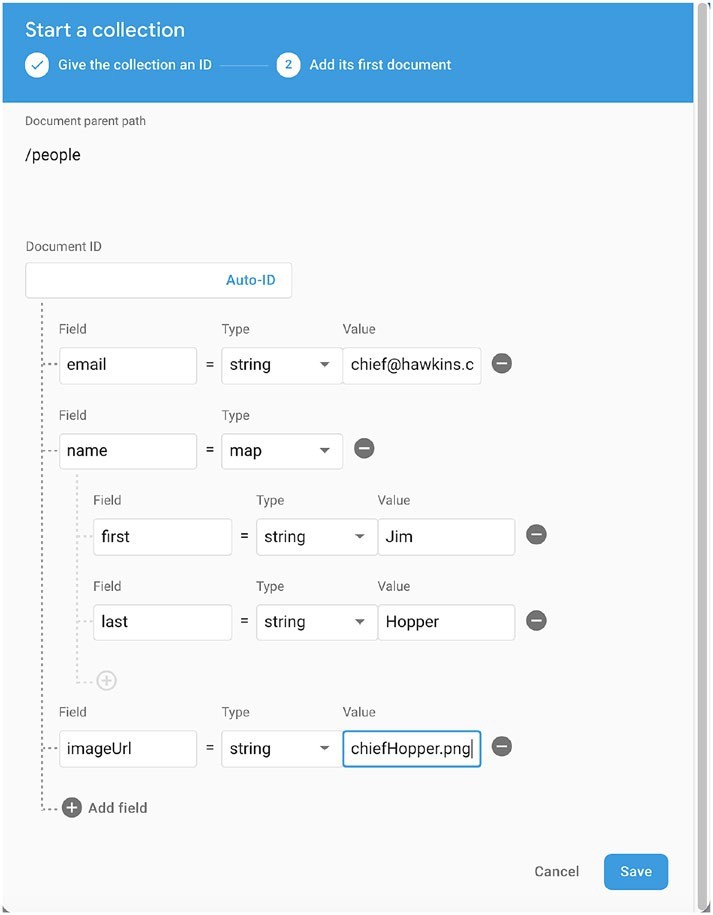
Sütunlar Alanlar

Bir NoSQL veritabanının anahtarları ve değerleri vardır, ancak bunların sabit bir yapısı yoktur. Başka bir deyişle, bir koleksiyondaki her belge aynı koleksiyondaki diğer belgelerden farklı alanlara sahip olabilir. Bu durum geleneksel veritabanları ile NoSQL veritabanları arasındaki en büyük farktır ve

en zor şeydir.

Şimdi bir koleksiyon oluşturacağız. Koleksiyonu Başlat'a basın ve ona bir kimlik verin.

Ardından bir veya daha fazla belge ekleyebileceksiniz (Şekil [12-5](#_bookmark8)).



***Şekil 12-5.*** *İlk belgenizi koleksiyona ekleme*

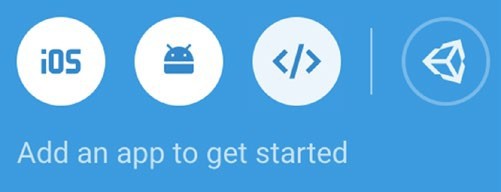
Artık yalnız belgenizi Cloud Firestore görüntüleyicide görebileceksiniz.

Buradan belge ekleyebilir, belgeleri silebilir ve belgeleri değiştirebilirsiniz.

Veritabanını doğrudan Firebase web sitesinden koruyabilmemiz güzel, ancak amacımız bunu uygulamamızdan yapmak. Bu yüzden iOS ve Android uygulamalarımızı Cloud Firestore'dan okuyacak şekilde yapılandırmalıyız.

## Bir iOS uygulaması oluşturma

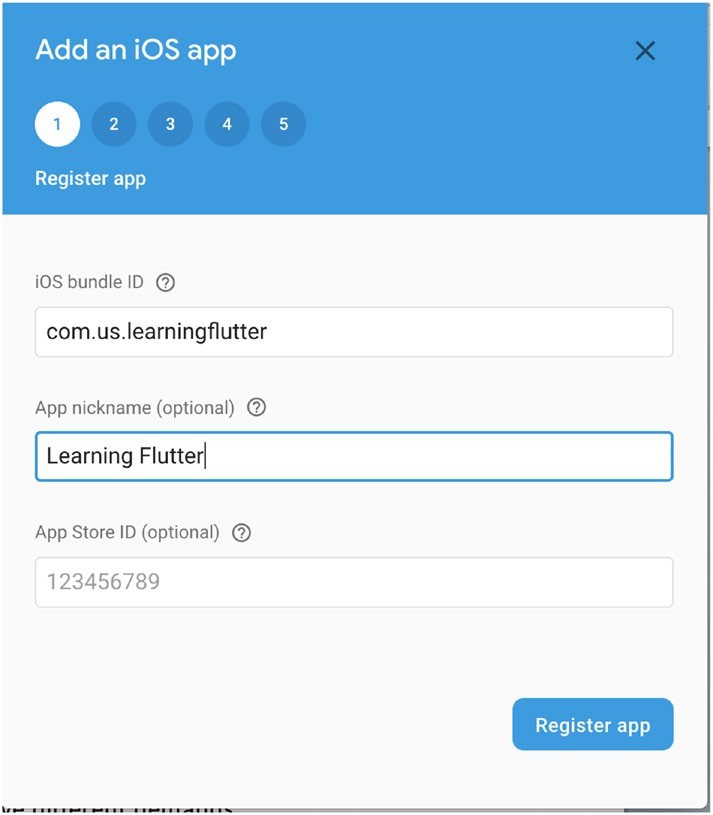
Projenizin genel bakış sayfasında, projenizde uygulamalar oluşturmak için düğmeler görmelisiniz (Şekil [12-6](#_bookmark9)).



***Şekil 12-6.*** *Firestore uygulamaları oluşturma seçenekleriniz*

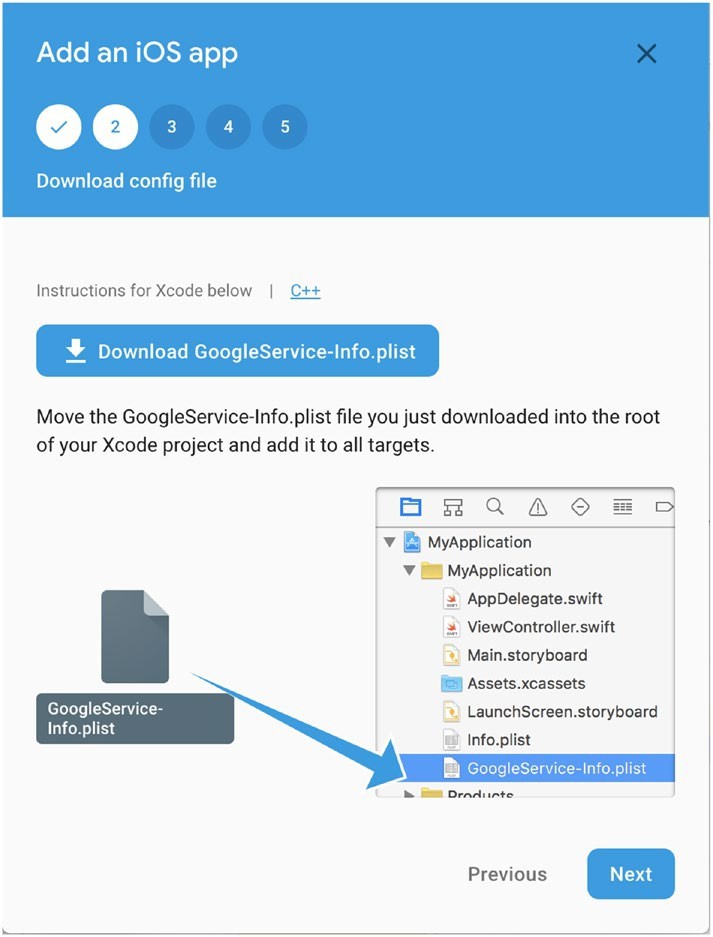
iOS düğmesine tıklayın.

Uygulama için bir ad girin (Şekil [12-7](#_bookmark10)).



***Şekil 12-7.*** *iOS uygulamanıza kimlik ve takma ad verme*

Uygulamayı kaydettikten sonra Şekil [12-8](#_bookmark11)'deki gibi bir görünüm göreceksiniz.



***Şekil 12-8.*** *Firestore, otomatik oluşturulan yapılandırma dosyasını*

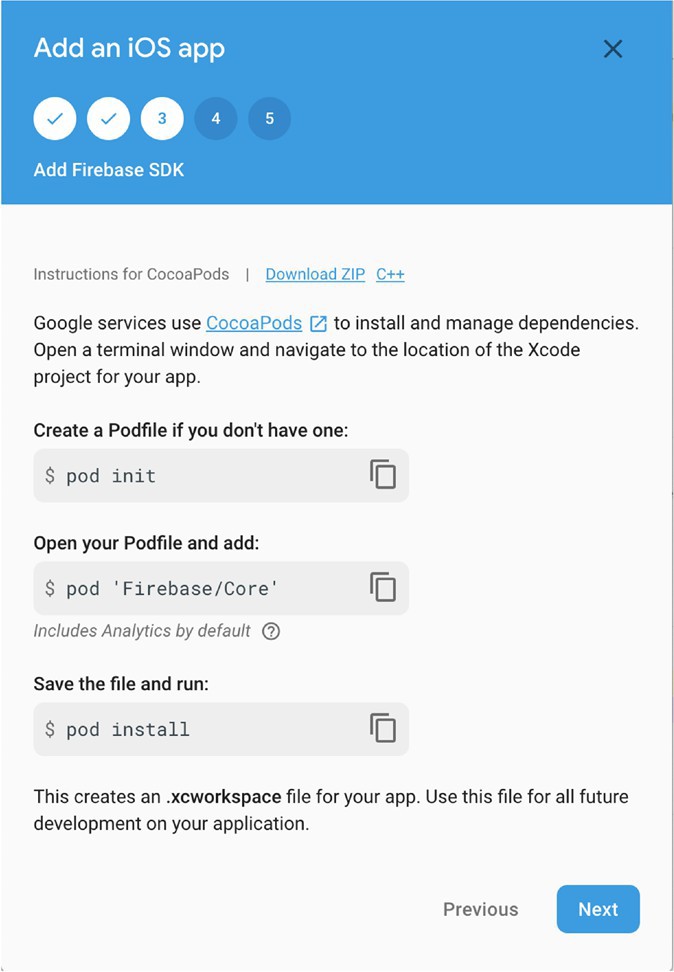
*indirmeniz için kullanılabilir hale getirir*

GoogleService-info.plist dosyasını indirin ve iOS/Runner/ Runner dizininde saklayın (Şekil [12-8](#_bookmark11)). Info.plist dosyasını gördüğünüzde doğru klasörde olduğunuzu anlayacaksınız. Size gösterdikleri diyagram dikkatinizi dağıtmasın. Farklı görünecektir çünkü bir projenin Xcode sürümünü gösteriyorlar, ancak siz bir Flutter projesinde çalışıyorsunuz.

Google tarafından bize sağlanan belirli araçları kullanırsak Firebase'e bağlanmak daha kolay olacaktır. Bu, indirilmeleri ve yüklenmeleri gerektiği anlamına gelir

iOS/xcode projemize ekleyin. iOS, bağımlılıkları yönetmek için CocoaPods kullanır.[3](#_bookmark12) Henüz bir Podfile'ımız yoksa bir Podfile oluşturmalıyız. Bir Podfile eklemek ve .xcworkspace dosyasını oluşturmak için aşağıdaki talimatları izleyin (Şekil [12-9](#_bookmark13)).

3 JavaScript uygulamaları için npm/package.json, .Net için NuGet, Ruby projeleri için RubyGems vb.



***Şekil 12-9.*** *Google size bir Podfile oluşturma adımlarını ve*

*.xcworkspace*