**Proje Yapısı :**

Proje ağacı:

* Lib
  + Models
    - kategori.dart
    - notlar.dart
  + utils
    - database\_helper.dart
  + kategori\_islemleri.dart
  + main.dart
  + not\_detay.dart

├───**lib**

│ ├───**models**

│ │ ├───kategori\_model.dart

│ │ └───not\_model.dart

│ ├───**utils**

│ │ └───database\_helper.dart

│ ├───kategori\_islemleri.dart

│ ├───main.dart

│ └───not\_detay.dart

**.**

**.**

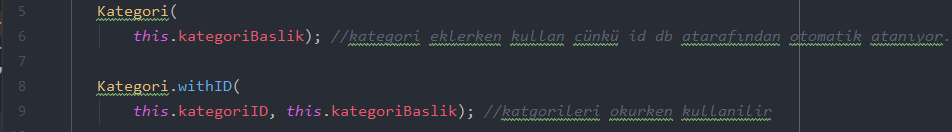
**models Package**

Kategori\_model.dart :

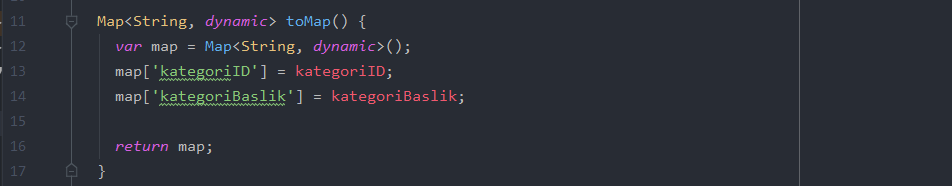
* iki adet özelliği bulunmakta.



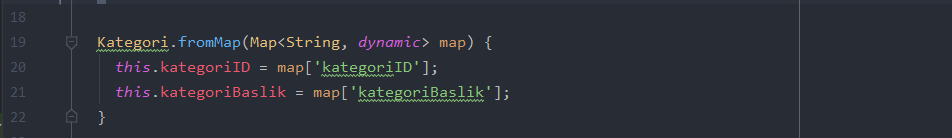
* 3(üç) adet yapıcı metodu bulunmaktadır.
* İlk metot kategori eklenirken kullanılıyor çünkü otomatik artan bir ID değeri bulunmakta.
* İkinci metot ise kategoriyi okurken kullanıyoruz.



* toMap() metodu = anahtarları String diğer değerleri ise değişebilen türden olan bir map yapısını oluşturan ve geri döndüren metottur.

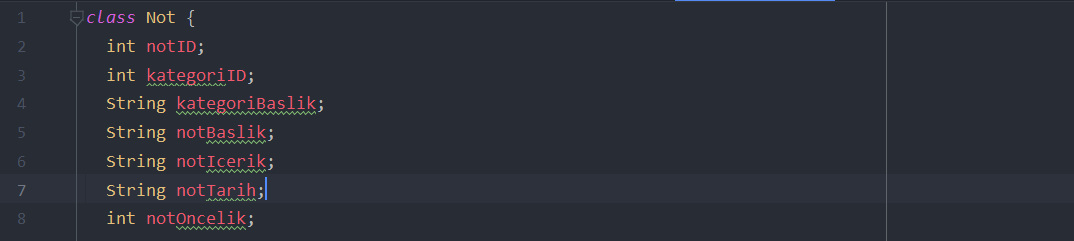


* fromMap() isimlendirilmiş yapıcı metodu = içerisine bir map değeri alan ve obje oluştururan metottur.

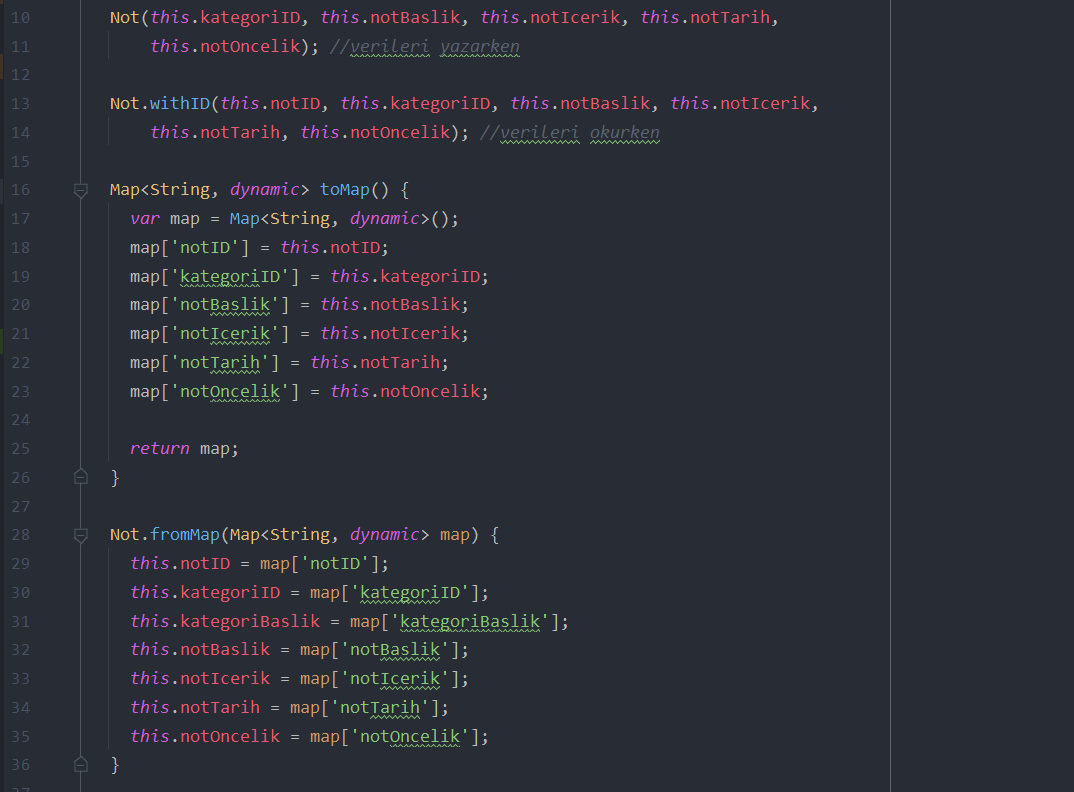


not\_model.dart :

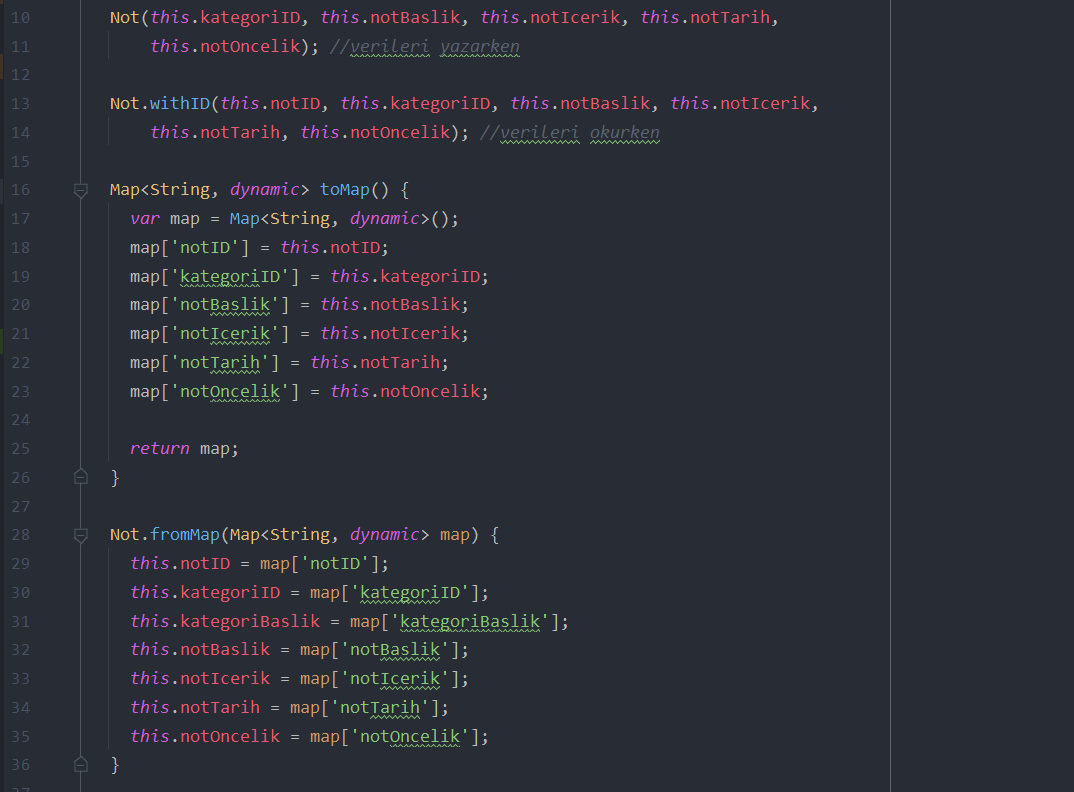
* 7 adet özelliği bulunmakta.



* 4(dört) adet yapıcı metodu bulunmakta.
* Not() yapıcı metodu verileri veri tabanına yazarken kullanılmakta çünkü ID değeri otomatik artan şekilde veri tabanı şemasında verilmiştir.
* Not.withID() yapıcı metodu ise verileri okurken kullanılmaktadır.



* toMap() metodu not nesnesini map olarak geri döndüren bir yapıya sahiptir.
* Not.fromMap() yapıcı metodu ise bir map nesnesi alarak bunu obje olarak üretir.



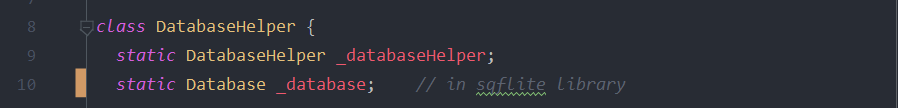
**utils Package**

database\_helper.dart:

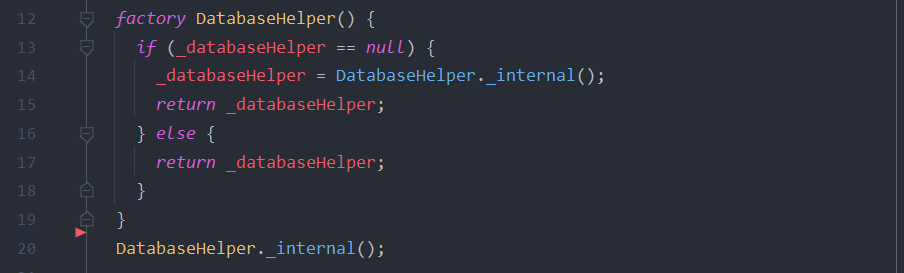
* Veritabanı olarak sqflite kullanıldığı için gerekli kütüphaneyi import ediyoruz.



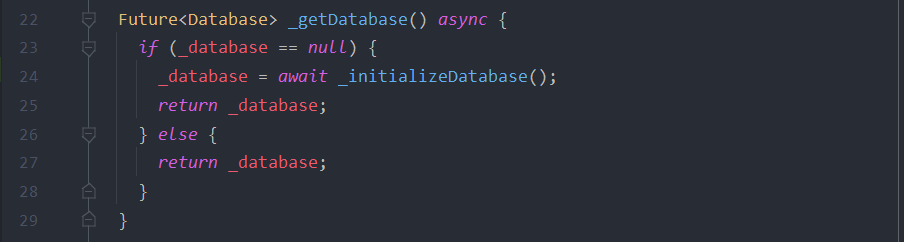
* Özelliklerin static tanımlanmasının sebebi nesneye özgü değil sınıfa özgü olmasını sağlamaktır.
* \_database nesnesi silme, ekleme, güncelleme gibi işlemler için.



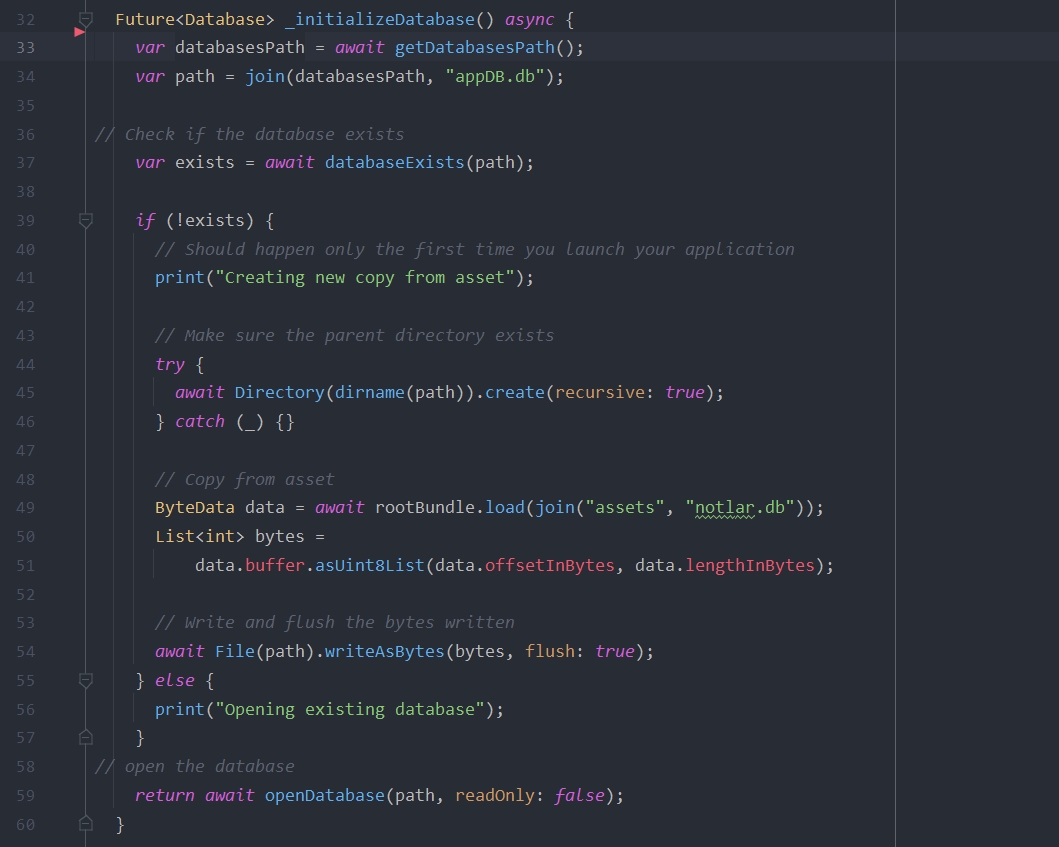
* Burada \_databaseHelper daha önce oluşturuldu mu onu kontrol ediyoruz.
* Oluşturulmamış ise DatabaseHelper.internal() diyerek oluşturuyoruz.
* Not: Eğer yapıcı metodunuzda return ifadesi var ise factory anahatarı kullanılmalıdır.



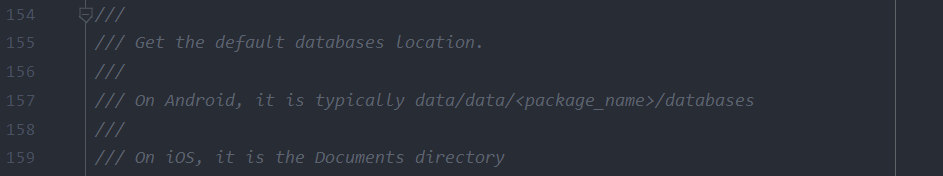
* \_getDatabase() metodunda ise database nesnesi boş değil ise return et, boş ise \_initializeDatabase(); metodunu ile veri tabanı oluşturma işlemi yapıyoruz.
* Uzun süren bir işlem olduğu için async olmalı ve await olmalıdır.



* Bu kod kütüphanenin issues kısmından alınıyor.
* Path alınıyor ve bu path de böyle bir veritabanı var mı kontrol ediliyor. Eğer yok ise yeni bir kopya oluşturuluyor. Var ise database geri döndürülmekte.



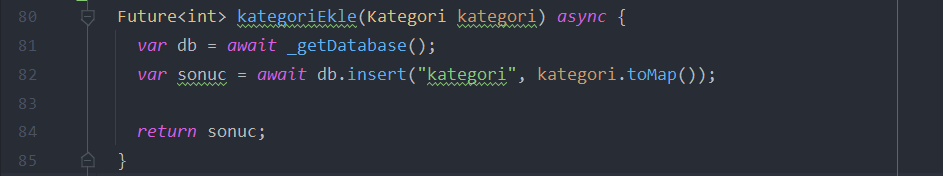
* Path android ve ios için aşağıdaki gibi yerleştirilmektedir.



* Uzun süren bir işlem o yüzden async.
* Db.query bize bir map listesi döndürür.



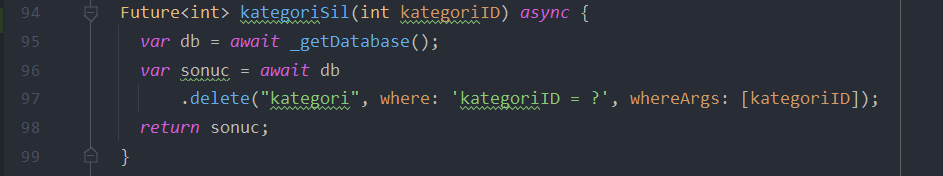
* kategoriEkle() metodu



* kategoriGuncelle() metodu



* kategoriSil() metodu



Not tablosu için de aynı şekil devam ediyor…