KONU: Flutter ile Firebase'de Resim Belgesi Saklama ve Geri Uygulamaya Çekme.

Bazı durumlarda flutter uygulamalarında kullanıcıdan fotoğraf alınması ve bunun bir databasede saklanması veya databasede bulunan bir fotoğrafın kullanıcıya gösterilmesi gerekebilir. Bu durumlar için firebase storage kullanılabilir. Firebase storage, arzuladığımız dosyaların firebasede tutulup gerektiğinde uygulamadan çekilmesini sağlayan bir araçtır. Kullanmak için firestoreu projelerimize eklerken yaptığımız işlemleri uygulamamız gerekir. Bir kez firebase projeye eklendi mi hem storage hem de firestore aynı projede kullanılabilir.

Firebase storagea upload yapmak için aşağıdaki fonksiyon kullanılabilir:

```
Future uploadImageToFirebase(BuildContext context) async {
    String fileName = basename(_imageFile!.path);
    firebase_storage.Reference ref = firebase_storage.FirebaseStorage.instance
        .ref()
        .child('uploads')
        .child('/$fileName');

    final metadata = firebase_storage.SettableMetadata(
        contentType: 'image/jpeg',
        customMetadata: {'picked-file-path': fileName});
    firebase_storage.UploadTask uploadTask;
    //late StorageUploadTask uploadTask = firebaseStorageRef.putFile(_imageFile);
    uploadTask = ref.putFile(io.File(_imageFile!.path)!, metadata);

    firebase_storage.UploadTask task = await Future.value(uploadTask);
    Future.value(uploadTask)
        .then((value) => {print("Upload file path ${value.ref.fullPath}")})
        .onError((error, stackTrace) =>
            {print("Upload file path error ${error.toString()} ")});
  }
```

Not: Firebase storage kullanmak için firebase_storage, firebase_core ve path_provider paketlerinin projeye eklenmesi gereklidir.

Firebase storagetan projeye dosya çekmek içinse iki farklı yöntem kullanılabilir.

1- Image.network():

```
2-  Future downloadFileExample() async {
3-      final ref = firebase_storage.FirebaseStorage.instance.ref().child('Richard-
    Feynman_s-Drawings-Untitled-2.jpg');
4-    // no need of the file extension, the name will do fine.
5-    url = await ref.getDownloadURL();
6-    setState(() {
7-
8-    });
9-    print(url);
10-
```

Yukarıdaki fonksiyonla dosya urlsi çekildikten sonra Image.network() widgetına url verilerek fotoğraf çekilebilir (aynı şekilde url kullanılarak diğer dosyalarla da işlem yapılabilir).

2- Fotoğraf widget özel olarak oluşturulup işlemler orada yapılabilir:

```
class FirebaseImage extends StatefulWidget {
  final String storagePath;

  FirebaseImage({
    required this.storagePath,
  }) : super(key: Key(storagePath));

  @override
  State<FirebaseImage> createState() => _FirebaseImageState();
}

class _FirebaseImageState extends State<FirebaseImage> {
  File? _file;

  @override
  void initState() {
    init();
    super.initState();
  }

  Future<void> init() async {
    final imageFile = await getImageFile();
    if (mounted) {
      setState(() {
        _file = imageFile;
      });
    }
  }

  Future<File?> getImageFile() async {
```

```
    final storagePath = widget.storagePath;
    final tempDir = await getTemporaryDirectory();
    final fileName = widget.storagePath.split('/').last;
    final file = File('${tempDir.path}/$fileName');

    // If the file do not exists try to download
    if (!file.existsSync()) {
      try {
        file.create(recursive: true);
        await FirebaseStorage.instance.ref(storagePath).writeToFile(file);
      } catch (e) {
        // If there is an error delete the created file
        await file.delete(recursive: true);
        return null;
      }
    }
    return file;
  }

  @override
  Widget build(BuildContext context) {
    if (_file == null) {
      return const Icon(Icons.error);
    }
    return Image.file(
      _file!,
      width: 100,
      height: 100,
    );
  }
}
```

Bu işlem aynı zamanda dosyaları indirmektedir ve zaten olan dosyaları tekrar indirmediği için birden fazla dosyayla uğraşılan ve tekrar eden durumlarda daha avantajlıdır.

Berke Can GÖKTAŞ

16.01.2022