

KFirestoreQueryWidget User Guide:

It is a re-usable flutter widgets to query and display data from the firestore to our flutter application. It is generic widgets so we have to pass type of the data to be displayed.

To use this widgets we have to first create a model class of the document of the collection and fromMap method to parse the data



```
import 'package:cloud_firestore/cloud_firestore.dart';

class User {
  final String id;
  final String name;
  final String email;
  final String gender;
  final DocumentSnapshot lastDoc;

  User({
    this.id,
    this.name,
    this.email,
    this.gender,
    this.lastDoc,
  });

  factory User.fromMap(
    Map<String, dynamic> map, String id, DocumentSnapshot lastDoc) {
    return User(
      id: id,
      name: map['name'],
      email: map['email'],
      gender: map['gender'],
      lastDoc: lastDoc,
    );
  }
}
```

And also we have firestore service to make firebase query easier


```
import 'package:cloud_firestore/cloud_firestore.dart';

class FirestoreService {
  static FirestoreService _service = FirestoreService._();
  FirestoreService._();
  factory FirestoreService() => _service;

  final _firestore = FirebaseFirestore.instance;

  Future<List<T>> queryData<T>({
    String path,
    T builder(Map<String, dynamic> data, String id, DocumentSnapshot lastDoc),
    Query queryBuilder(Query query),
  }) async {
    Query query = _firestore.collection(path);
    if (queryBuilder != null) {
      query = queryBuilder(query);
    }
    final _doc = await query.get();
    return _doc.docs
      .map<T>((e) => builder(e.data(), e.id, _doc.docs.last))
      .toList();
  }
}
```

We have to pass the collection path and also the builder to parse the data
And also we can pass firestore query to queryBuilder parameter as



```
KFirestoreQueryWidget<Job>(  
  perPage: 15,  
  path: 'jobs',  
  builder: (data, id, lastDoc) => Job.fromMap(data, id, lastDoc),  
  doc: (List<Job> items) => items.last?.lastDoc,  
  queryBuilder: (query) => query.where('status', isEqualTo: false),  
  itemBuilder: (BuildContext context, Job item) {  
    return Card(  
      child: ListTile(  
        title: Text(item.title),  
        subtitle: Text(item.status.toString()),  
      ),  
    );  
  },  
);
```