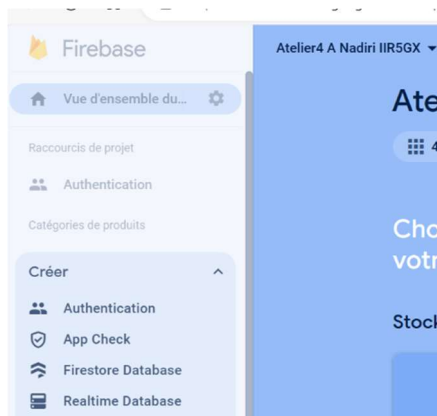


Atelier 4-2 - Firebase - Firestore

Cloud Fire est une base de données NoSQL orientée documents.

Créer une nouvelle base de données Firestore

1. A partir de la console firebase (<https://console.firebase.google.com/>), ouvrez le projet créé dans l'atelier précédent, et cliquez sur l'élément « Firebase Database » du menu « Créer »



2. Cliquez sur le bouton « Créer une base de données »



3. Sélectionnez éventuellement un nouvel emplacement et cliquez sur Next

A screenshot of the 'Créer une base de données' (Create database) wizard in the Firebase console. The wizard has two steps: '1 Définir le nom et l'emplacement' (Define name and location) and '2 Fixer les règles' (Set rules). The first step is active. It shows a text field for 'ID de la base de données' (Database ID) with the value '(default)'. Below it is a dropdown menu for 'Emplacement' (Location) with 'nam5 (United States)' selected. A warning message states: 'Une fois défini, cet emplacement ne peut plus être modifié. Ce paramètre s'appliquera aussi à votre bucket Cloud Storage par défaut.' (Once defined, this location cannot be modified. This parameter will also apply to your default Cloud Storage bucket.) At the bottom, there are 'Annuler' (Cancel) and 'Suivant' (Next) buttons.

4. Sélectionnez l'option « Démarrer en mode test » et cliquez sur « Activer »

Créer une base de données

1 Définir le nom et l'emplacement 2 Fixer les règles

Après avoir défini votre structure de données, vous devez spécifier des règles pour sécuriser vos données.
[En savoir plus](#)

☐ Démarrer en mode de production
 Par défaut, vos données sont publiques pour permettre une configuration rapide. Toutefois, vous devez modifier vos règles de sécurité dans les 30 jours pour autoriser l'accès client en lecture/écriture sur le long terme.

☒ Démarrer en mode test
 Par défaut, les règles de sécurité en mode test autorisent tout utilisateur disposant de la référence de votre base de données à afficher, modifier et supprimer toutes les données qu'elle contient pendant les 30 prochains jours.

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /(document=**) {
      allow read, write: if
        request.time < timestamp.date(2023, 12, 19);
    }
  }
}
```

Si vous activez Cloud Firestore, vous ne pourrez pas utiliser Cloud Datastore avec ce projet

Annuler **Activer**

5. Cliquez sur « Commencer une collection », nommez la collection « produits » et cliquez sur suivant.

Commencer une collection

1 Spécifier un ID pour la collection 2 Ajouter le premier document

Chemin parent
/

ID de collection ⓘ
produits

Annuler **Suivant**

6. Pour ajouter un produit (une collection Firestore doit contenir au moins un document), cliquez sur « id généré automatiquement » et ajoutez les informations du produit (Il n'est pas obligatoire d'ajouter les mêmes valeurs que dans la figure ci-dessus), et enfin cliquez sur « Enregistrer »

Commencer une collection

1 Spécifier un ID pour la collection 2 Ajouter le premier document

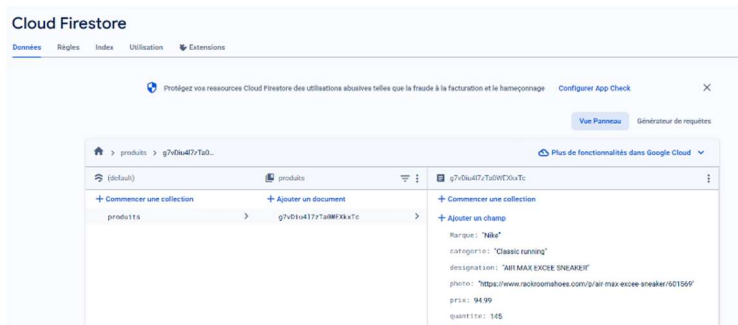
Chemin parent du document
/produits

ID du document ⓘ
g7vDiu4I7zTa0WEXkxTc

Champ	Type	Valeur
designation	string	AIR MAX EXCEE
prix	number	94.99
quantite	number	145
photo	string	https://www.rack
categorie	string	Classic running
Marque	string	Nike

Annuler **Enregistrer**

Résultat après création de la collection « produits »



Configuration du projet flutter

1. Ajoutez dans le projet flutter « atelier4 ... » le package cloud_firestore

Liste des produits

1. Ajoutez le widget statefull ListeProduits ()
2. Ajoutez la classe Produit (produit.dart)

```
class Produit {
  String id;
  String marque;
  String designation;
  String categorie;
  double prix;
  String photo;
  int quantite;

  Produit({
    required this.id,
    required this.marque,
    required this.designation,
    required this.categorie,
    required this.prix,
    required this.photo,
    required this.quantite,
  });
}
```

3. Ajoutez dans la classe Produit un constructeur nommé fromFirestore de type factory dont le rôle est de créer un objet Produit à partir d'un document Firestore.

```
factory Produit.fromFirestore(DocumentSnapshot doc) {
  Map data = doc.data() as Map;
  return Produit(
    id: doc.id,
    marque: data['marque'] ?? '',
    designation: data['designation'] ?? '',
    categorie: data['categorie'] ?? '',
    prix: (data['prix'] ?? 0.0).toDouble(),
    photo: data['photo'] ?? '',
    quantite: data['quantite'] ?? 0,
  );
}
```

4. Dans la classe privée `_ListeProduitsState`, ajoutez une référence vers la base de données

```
FirebaseFirestore db = FirebaseFirestore.instance;
```

5. Remplacez `Placeholder` par `Scaffold`, et définir la propriété `appBar`

```
Widget build(BuildContext context) {  
  return Scaffold(  
    body: StreamBuilder<QuerySnapshot>(  
      stream: db.collection('produits').snapshots(),  
      builder: (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {  
        if (snapshot.hasError) {  
          return const Center(child: Text('Une erreur est survenue'));  
        }  
  
        if (snapshot.connectionState == ConnectionState.waiting) {  
          return const Center(  
            child: CircularProgressIndicator(),  
          ); // Center  
        }  
  
        List<Produit> produits = snapshot.data!.docs.map((doc) {  
          return Produit.fromFirestore(doc);  
        }).toList();  
  
        return ListView.builder(  
          itemCount: produits.length,  
          itemBuilder: (context, index) => ProduitItem(  
            produit: produits[index],  
          ), // ProduitItem  
        ); // ListView.builder  
      },  
    ), // StreamBuilder  
  ); // Scaffold  
}
```

6. Ajoutez un `StreamBuilder` de type `QuerySnapshot` dans la propriété `body`

7. Ajoutez le widget `ProduitItem`, exemple

```
class ProduitItem extends StatelessWidget {  
  ProduitItem({Key? key, required this.produit}) : super(key: key);  
  
  final Produit produit;  
  
  @override  
  Widget build(BuildContext context) {  
    return ListTile(  
      title: Text(produit.designation),  
      subtitle: Text(produit.marque),  
      trailing: Text('${produit.prix} €'),  
    ); // ListTile  
  }  
}
```

Pour tester le widget `ListeProduits`, vous pouvez ajouter la fonction `main` à la fin du fichier

```

Run | Debug | Profile
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);
  runApp(const MaterialApp(
    home: ListProduits(),
  )); // MaterialApp
}

```

Configuration de l'application Android

1. Dans le fichier Android/local.properties, définir la version minimale à 24

```

sdk.dir=C:\AppData\AndroidSDK
flutter.sdk=C:\Tools\flutter
flutter.buildMode=debug
flutter.versionName=0.1.0
flutter.minSdkVersion=24

```

2. Dans le fichier Android/app/build.gradle, ajoutez multiDexEnabled = true, dans la section defaultConfig

```

defaultConfig {
  minSdkVersion flutter.minSdkVersion
  targetSdkVersion flutter.targetSdkVersion
  versionCode flutterVersionCode.toInteger()
  versionName flutterVersionName
  multiDexEnabled = true
}

```

Exécution de l'application

1. Exécuter et tester l'application dans Android.
2. Dans la console firebase, changer les règles d'accès à la base de données produits (uniquement les utilisateurs authentifiés doivent avoir accès à la base de données)

Cloud Firestore

Données **Règles** Index Utilisation Extensions

Développer et tester

À l'instant
modifications non publiées

nov. 19, 2023 • 6:02 PM

modifications non publiées | Publier | Supprimer

```

1 rules_version = '2';
2
3 service cloud.firestore {
4   match /databases/{database}/documents {
5     match /(document=*) {
6       allow read, write: if request.auth.uid != null;
7     }
8   }
9 }

```

3. Exécutez à nouveau l'application (Redémarrer l'émulateur), normalement la base de données de données ne sera plus accessible sans authentification.

Activation de l'authentification

1. Supprimez la méthode main dans le fichier liste_produits.dart
2. Modifiez LoginEcran comme suit :

```
@override
Widget build(BuildContext context) {
  return StreamBuilder<User?>(
    stream: FirebaseAuth.instance.authStateChanges(),
    builder: (context, snapshot) {
      if (!snapshot.hasData) {
        return SignInScreen();
      }

      return ListProduits();
    },
  ); // StreamBuilder
}
```

Exercice

Améliorez la présentation de la liste des produits

Autres opérations sur les collections firestore

Suppression

```
db.collection('produits').doc(produitId).delete();
```

Ajout

```
Firestore.instance.collection('produits').add({
  'marque': "Hello",
  'designation': "Classic",
  'categorie': 'cat',
  'prix': 55,
  'photoUrl': "",
  'quantite': 155
});
```

On peut aussi ajouter un constructeur nommé toJson et écrire

```
db.collection('produits').add(p.toJson());
```