



Atelier 9

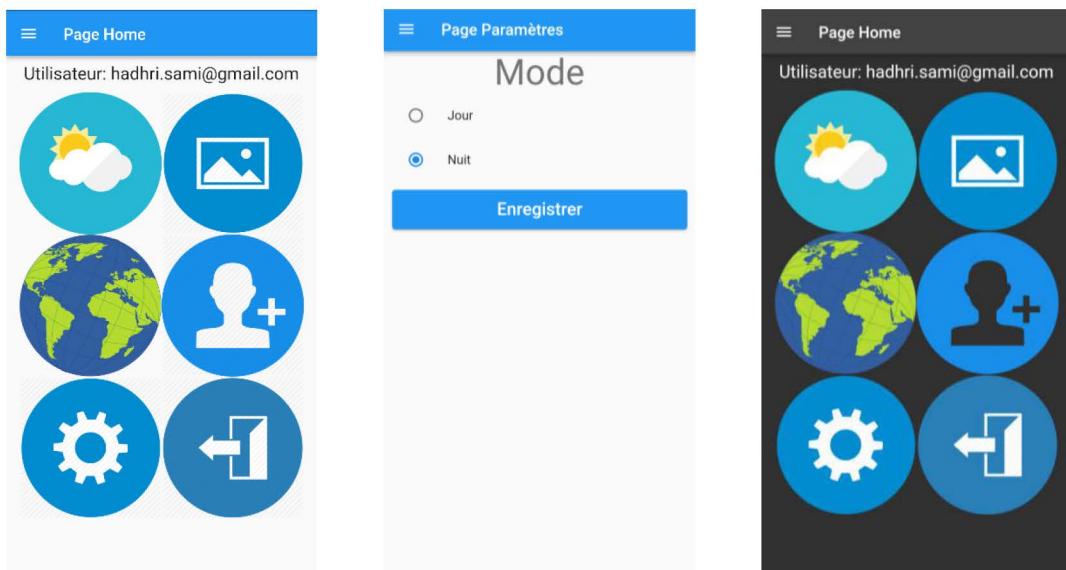
Pages Authentification, Inscription & Paramètres

Matière : ATELIER FRAMEWORK CROSS-PLATFORM

DSI3

Enseignants : S. Hadhri & M. Hadjji

L'objectif de ce TP est d'intégrer firebase dans notre application « voyage ». Nous allons utiliser le service authentification pour gérer les utilisateurs dans le cloud au lieu des préférences partagées et le service database pour sauvegarder dans la base de données en nuages le mode de l'application (clair ou sombre).



Etape 1 : Présentation & configuration de Firebase

Firebase est une plate-forme fournie par Google pour activer votre application mobile. Firebase regorge de fonctionnalités importantes telles que l'authentification des utilisateurs et la messagerie cloud.

Dans cet atelier, nous allons utiliser le service d'authentification Firebase dans notre application Flutter.

L'authentification des utilisateurs est une fonctionnalité essentielle dans presque toutes les applications Web ou mobiles. Il sécurise les données pertinentes, permet l'accès aux utilisateurs autorisés.

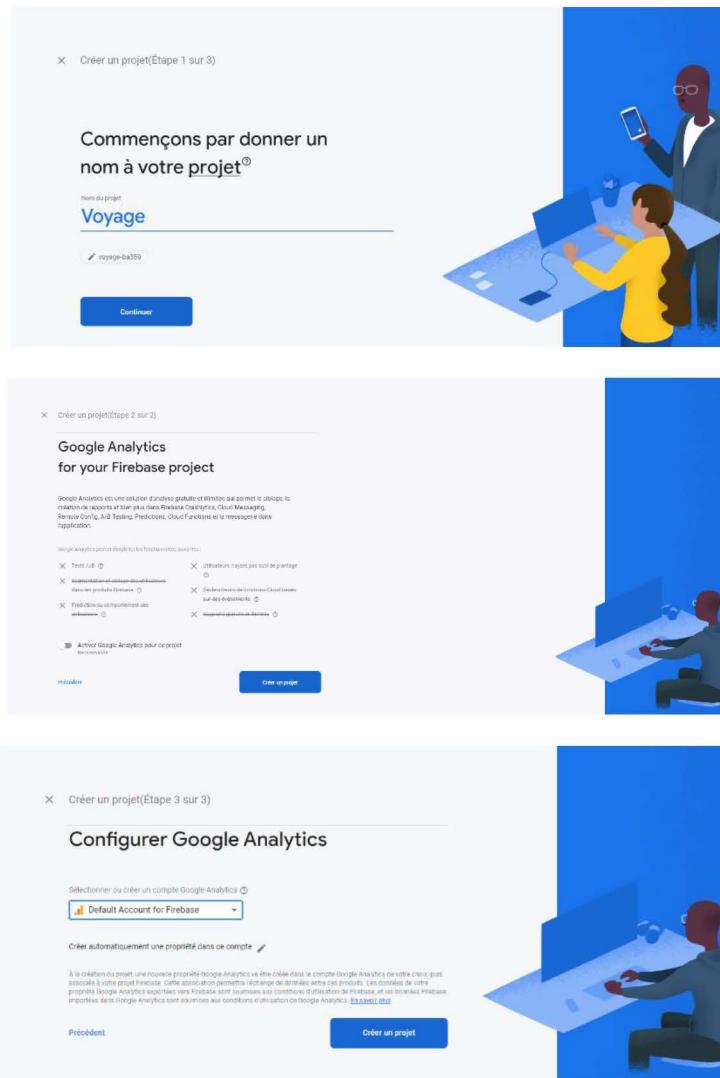


1. Configuration du projet Firebase

- Pour configurer un compte, accéder à console.firebaseio.google.com, puis cliquer sur « Ajouter un projet ».



- Saisir le nom du projet et cliquer sur « Créer un projet ».



The image contains three vertically stacked screenshots of the Firebase project creation process:

- Step 1: Create a project (Step 1 of 3)**

Commençons par donner un nom à votre projet[®]

Name of project: Voyage

Checkboxes: voyage-ba359, voyage-ba359

Buttons: Continuer (Continue) and Annuler (Cancel)
- Step 2: Google Analytics for your Firebase project (Step 2 of 3)**

Google Analytics for your Firebase project

Information about Google Analytics: "Google Analytics est une solution Firebase globale et native qui permet de débloquer l'accès à des données utiles et précises via les trois produits : Tag Manager, Cloud Messaging, Remote Config, A/B Testing, Predictions, Cloud Functions et la messagerie dans les applications."

Options: Test / Développement et collecte de données, Analytics et collecte de données, Analytics et collecte de données, Predictions et collecte de données, et Mesures et collecte de données.

Checkboxes: Ajouter Google Analytics pour ce projet (checkbox checked), Recommandé

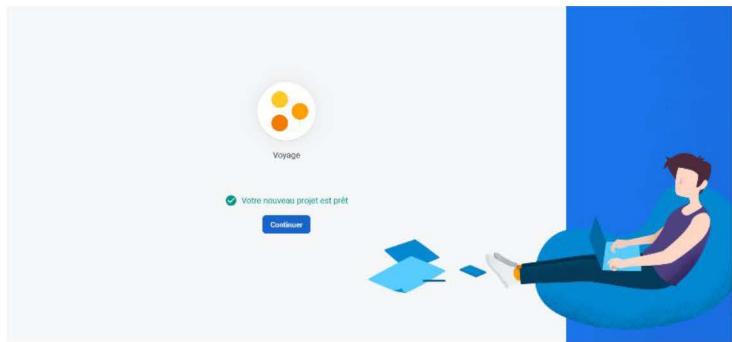
Buttons: Précédent (Previous) and Crée un projet (Create project)
- Step 3: Configure Google Analytics (Step 3 of 3)**

Configurer Google Analytics

Select or create a Google Analytics account: Default Account for Firebase

Information: "À la création du projet, une nouvelle propriété Google Analytics est créée dans le compte Google Analytics à votre choix, puis associée à votre projet Firebase. Cette association permettra l'échange de données entre ces deux produits. Les données de cette propriété Google Analytics associée avec Firebase sont soumises aux conditions d'utilisation de Firebase, et les données Firebase intégrées dans Google Analytics sont soumises aux conditions d'utilisation de Google Analytics. [En savoir plus](#)"

Buttons: Précédent (Previous) and Crée un projet (Create project)



2. Une fois le projet est créé, le tableau de bord Firebase sera affiché permettant d'ajouter Firebase à une application iOS, Android, Web, Unity ou bien Flutter.: Cliquer sur l'icône Flutter.



a. Préparer l'espace de travail

- Installer la CLI Firebase :
 - Installer [Node.js](#)
 - Exécuter la commande
`npm install -g firebase-tools`
- Se connecter à Firebase à l'aide du compte Google
 - Exécuter la commande suivante :
`firebase login`
 - ➔ Cette commande ouvre une page Web et connecte la machine locale à Firebase et donne accès aux projets Firebase.

```
Woohoo!
Firebase CLI Login Successful
You are logged in to the Firebase Command-Line
interface. You can immediately close this window and
continue using the CLI.
```

- Vérifier que la CLI est correctement installée et accède au compte en répertoriant les projets Firebase
 - Exécuter la commande suivante :
`firebase projects:list`

b. Installer et exécuter la CLI FlutterFire

- Depuis n'importe quel répertoire, exécuter cette commande :
`dart pub global activate flutterfire_cli`
- Ensuite, à la racine du répertoire du projet Flutter, exécuter cette commande :
`flutterfire configure --project=YOUR_PROJECT_ID`
- ➔ YOUR_PROJECT_ID est l'ID de votre projet définit dans firebase. Cela enregistre automatiquement l'application par plate-forme auprès de Firebase et ajoute un fichier de configuration « lib.firebaseio_options.dart » au projet Flutter.

NB:

Vous devez ajouter le dossier « C:\Users*username*\AppData\Local\Pub\Cache\bin » dans le path (variables d'environnement)

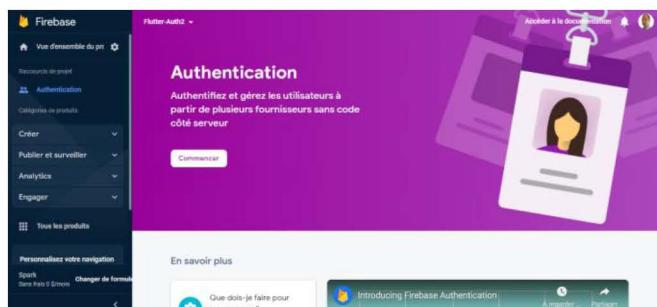
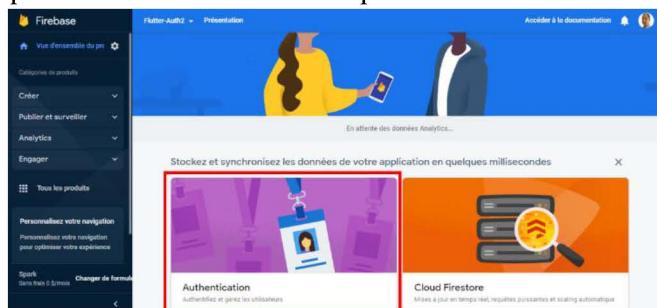
c. Initialiser Firebase et ajouter les plugins nécessaires

- Dans le fichier « pubspec.yaml », ajouter le package « firebase_core »:
https://pub.dev/packages/firebase_core

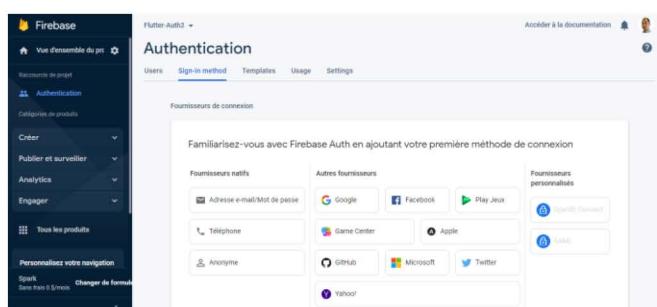
Etape 2 : Intégration de « firebase authentication »

3. Activation du service d'authentification :

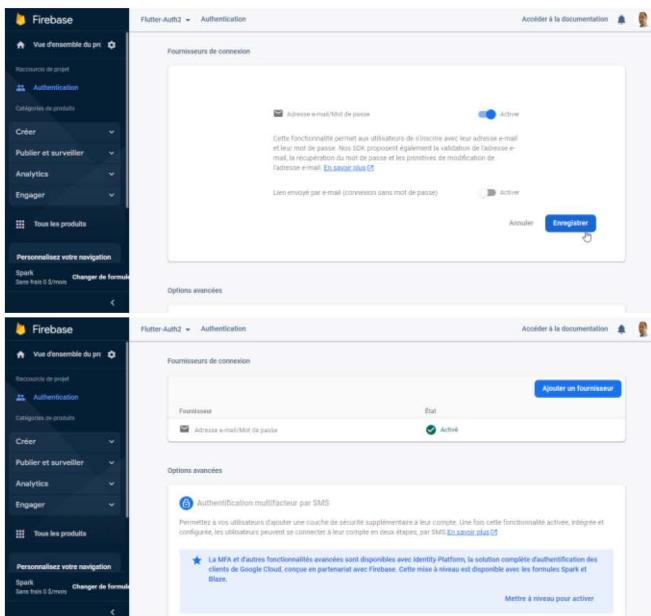
- Dans le menu de gauche, cliquer sur l'icône « Tous les produits » 
- Dans la console, cliquer sur « Authentication » puis sur « Commencer »



- Dans la fenêtre qui s'affiche, choisir « adresse e-mail / mot de passe » comme méthode de connexion.



- L'activer puis cliquer sur « Enregistrer »



4. Dans le fichier « pubspec.yaml », ajouter le package « firebase_auth »:
https://pub.dev/packages/firebase_auth

5. Modifier la méthode privée _onInscrire() dans la page d'inscription en utilisant Firebase plutôt que les préférences partagées.

Fichier inscription.page.dart

```

import 'package:firebase_auth/firebase_auth.dart';
...
Future<void> _onInscrire(BuildContext context) async {
  if (!txt_login.text.isEmpty && !txt_password.text.isEmpty) {
    try {
      await FirebaseAuth.instance.createUserWithEmailAndPassword(
        email: txt_login.text.trim(), password: txt_password.text.trim());
      Navigator.pop(context);
      Navigator.pushNamed(context, '/home');
    } on FirebaseAuthException catch (e) {
      SnackBar snackBar = SnackBar(content: Text(""));

      if (e.code == 'weak-password') {
        snackBar = SnackBar(
          content: Text('Mot de passe faible'),
        );
      } else if (e.code == 'email-already-in-use') {
        snackBar = SnackBar(
          content: Text('Email déjà existant'),
        );
      }
      ScaffoldMessenger.of(context).showSnackBar(snackBar);
    }
  } else {
    const snackBar = SnackBar(
      content: Text('Id ou mot de passe vides'),
    );
    ScaffoldMessenger.of(context).showSnackBar(snackBar);
  }
}

```

Mot de passe faible Email déjà existant

- 6.Modifier le fichier « main.dart » pour tenir compte des changements effectués.

Fichier main.dart

```

import 'package:firebase_core/firebase_core.dart';

```

```

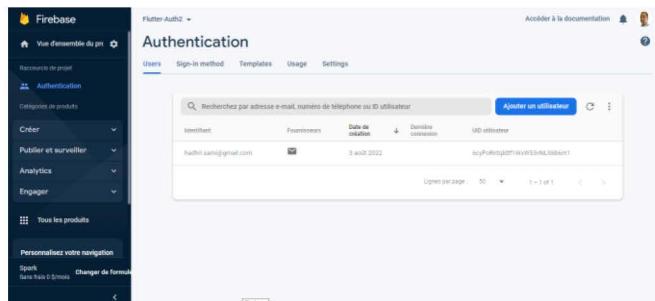
import 'package:firebase_auth/firebase_auth.dart';
import 'firebase_options.dart';
...
void main()=>runApp(MyApp());

Future<void> main() async{
    WidgetsFlutterBinding.ensureInitialized();
    await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);
    runApp(MyApp());
}

class MyApp extends StatelessWidget {
...
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            debugShowCheckedModeBanner: false,
            routes: routes,
            home: StreamBuilder<User?>(
                stream: FirebaseAuth.instance.authStateChanges(),
                builder: (context,snapshot){
                    if(snapshot.hasData)
                        return HomePage();
                    else
                        return AuthentificationPage();
                }
            ),
        );
    }
}

```

Puis, tester la nouvelle méthode d'authentification...



7. Modifier la page Home pour afficher l'utilisateur connecté. Changer la méthode privée `_Deconnexion()` en utilisant Firebase plutôt que les préférences partagées.Modifier également la déconnexion dans le fichier « `drawer.widget.dart` ». Tester la déconnexion.

Fichier `home.page.dart`

```

import 'package:firebase_auth/firebase_auth.dart';
...
class HomePage extends StatelessWidget {
    late SharedPreferences pref;
    ...
    @override
    Widget build(BuildContext context) {
        final user=FirebaseAuth.instance.currentUser;
        return Scaffold(
            drawer: MyDrawer(),
            appBar: AppBar(title: Text('Page Home')),
            body: Column(
                children: [

```

```

        Container(
            padding: EdgeInsets.all(10),
            child: Text("Utilisateur: ${user?.email}", style:
        TextStyle(fontSize: 22)),
    ),
    Center(
        child: Wrap(
            children: [
                ...GlobalParams.accueil as List).map((item) {
            return InkWell(
                child: Ink.image(
                    height: 180,
                    width: 180,
                    image: item['image'],
                ),
                onTap: () {
                    if ('${item["route"]}' != "/authentification")
                        Navigator.pushNamed(context, "${item['route']}");
                    else
                        _Deconnexion(context);
                },
            );
        ],
    ),
),
]
)
);
}
}

Future<void> _Deconnexion(context) async {
prefs = await SharedPreferences.getInstance(),
prefs.setBool("connecte", false),
FirebaseAuth.instance.signOut();
Navigator.of(context).pushNamedAndRemoveUntil(
    '/authentification', (Route<dynamic> route) => false);
}
}
}

```

Fichier drawer.widget.dart

```

import 'package:firebase_auth/firebase_auth.dart';
...
else
{
    prefs = await SharedPreferences.getInstance(),
    prefs.setBool("connecte", false),
    FirebaseAuth.instance.signOut();
    Navigator.of(context).pushNamedAndRemoveUntil('/authentification',
(Route<dynamic> route) => false);
}
...

```



8. Modifier la méthode privée `_onAuthentifier()` dans la page d'authentification en utilisant Firebase plutôt que les préférences partagées. Tester la connexion d'un utilisateur existant.

Fichier authentication.page.dart

```
import 'package:firebase_auth/firebase_auth.dart';
...
Future<void> _onAuthentifier(BuildContext context) async {
    try {
        await FirebaseAuth.instance.signInWithEmailAndPassword(
            email: txt_login.text.trim(), password: txt_password.text.trim());
        Navigator.pop(context);
        Navigator.pushNamed(context, '/home');
    } on FirebaseAuthException catch (e) {
        SnackBar snackBar= SnackBar(content: Text(""));

        if (e.code == 'user-not-found') {
            snackBar = SnackBar(
                content: Text('Utilisateur inexistant'),
            );
        } else if (e.code == 'wrong-password') {
            snackBar = SnackBar(
                content: Text('Vérifier votre mot de passe'),
            );
        }
        ScaffoldMessenger.of(context).showSnackBar(snackBar);
    }
}
```

Utilisateur inexistant

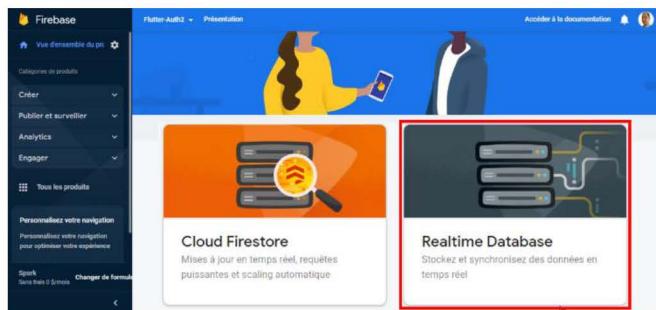
Vérifier votre mot de passe

Etape 3 : Intégration de « firebase realtime database »

9. Activation du service base de données :

- Dans le menu de gauche, cliquer sur l'icône « Tous les produits »
- Dans la console, cliquer sur « Realtime Database »





- Cliquer sur « Créer une base de données » → « Suivant » → « Activer »



- Cliquer sur l'onglet « Règles », modifier les règles de sécurité lecture/écriture à true puis cliquer sur « Publier »



- Revenir dans l'onglet « Données » et remarquer que la base de données est vide



10. Dans le fichier « pubspec.yaml », ajouter les packages suivants :

- firebase_database:

https://pub.dev/packages/firebase_database

11. Dans le fichier « android/app/build.gradle » du module app, ajouter :

Fichier build.gradle du module app

```
defaultConfig {
    // TODO: Specify your own unique Application ID
    // (https://developer.android.com/studio/build/application-id.html).
    applicationId "com.exemple.voyage"
    minSdkVersion flutter.minSdkVersion 19
```

```

        targetSdkVersion flutter.targetSdkVersion
        versionCode flutterVersionCode.toInt()
        versionName flutterVersionName
    }

```

12. Convertir la page paramètres en StatefulWidget puis créer 2 boutons radio (Jour et Nuit) et un bouton « Enregistrer ». Au clic sur ce bouton, le mode sera enregistré dans la base de données firebase.

Fichier parametres.page.dart

```

import 'package:flutter/material.dart';
import '../menu/drawer.widget.dart';
import 'package:firebase_database.firebaseio_database.dart';

String mode="Jour";
FirebaseDatabase fire = FirebaseDatabase.instance;
DatabaseReference ref = fire.ref();

class ParametersPage extends StatefulWidget {
    @override
    State<ParametersPage> createState() => _ParametersPageState();
}

class _ParametersPageState extends State<ParametersPage> {

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            drawer: MyDrawer(),
            appBar: AppBar(
                title: Text('Page Paramètres'),
            ),
            body: Column(
                mainAxisAlignment: MainAxisAlignment.start,
                children: [
                    Text('Mode', style: Theme.of(context).textTheme.headline3),
                    Column(
                        children: <Widget>[
                            ListTile(
                                title: const Text('Jour'),
                                leading: Radio<String>(
                                    value: "Jour",
                                    groupValue: mode,
                                    onChanged: (value) {
                                        setState(() {
                                            mode=value!;
                                        });
                                    }
                                ),
                            ),
                            ListTile(
                                title: const Text('Nuit'),
                                leading: Radio<String>(
                                    value: "Nuit",
                                    groupValue: mode,
                                    onChanged: (value) {
                                        setState(() {
                                            mode=value!;
                                        });
                                    }
                                ),
                            ),
                        ],
                    ),
                ],
            ),
        );
    }
}

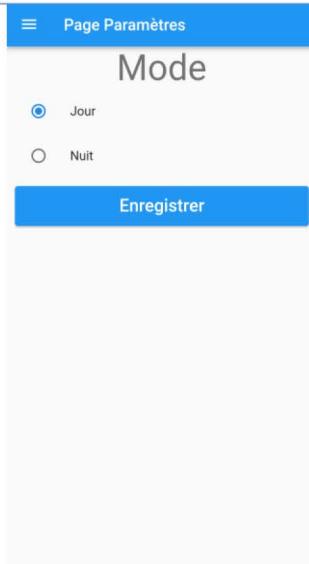
```

```

        ],
),
Container(
    padding: EdgeInsets.all(10),
    child: ElevatedButton(
        style: ElevatedButton.styleFrom(
            minimumSize: const Size.fromHeight(50)),
        onPressed: () {
            onSaveMode();
        },
        child: Text('Enregistrer', style: TextStyle(fontSize: 22))),
),
),
),
);
}

_onSaveMode() async {
    await ref.set({"mode": mode});
    print("mode changé");
    Navigator.pop(context);
}
}

```



13. Apporter les modifications nécessaires au fichier « main.dart » afin de charger le mode au démarrage de l’application à partir de firebase et de l’appliquer.

Fichier main.dart	
...	
ThemeData theme = ThemeData.light();	
Future<void> main() async {	
WidgetsFlutterBinding.ensureInitialized();	
await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);	
await _onGetMode();	
theme = (await _onGetMode() == "Jour") ? ThemeData.light() :	
ThemeData.dark();	
runApp(MyApp());	
}	
class MyApp extends StatelessWidget {	
final routes = {	
'/home': (context) => HomePage(),	

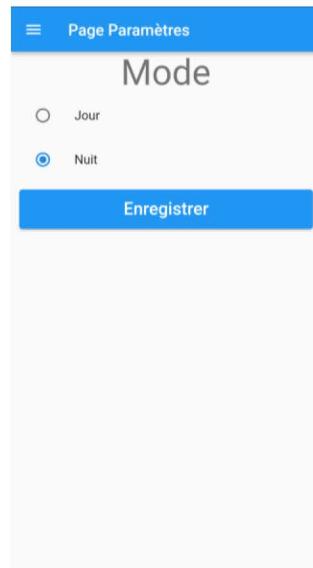
```

'authentication': (context) => AuthenticationPage(),
'inscription': (context) => InscriptionPage(),
'meteo': (context) => MeteoPage(),
'gallerie': (context) => GalleryPage(),
'pays': (context) => PaysPage(),
'/contact': (context) => ContactPage(),
'/parametres': (context) => ParametersPage(),
};

@Override
Widget build(BuildContext context) {
    return MaterialApp(
        debugShowCheckedModeBanner: false,
        routes: routes,
        theme: theme,
        home: StreamBuilder<User?>(
            stream: FirebaseAuth.instance.authStateChanges(),
            builder: (context, snapshot) {
                if (snapshot.hasData)
                    return HomePage();
                else
                    return AuthenticationPage();
            },
        ),
    );
}

Future<String> _onGetMode() async {
    final snapshot = await ref.child('mode').get();
    if (snapshot.exists)
        mode = snapshot.value.toString();
    else
        mode = "Jour";
    print(mode);
    return(mode);
}
}

```



Realtime Database

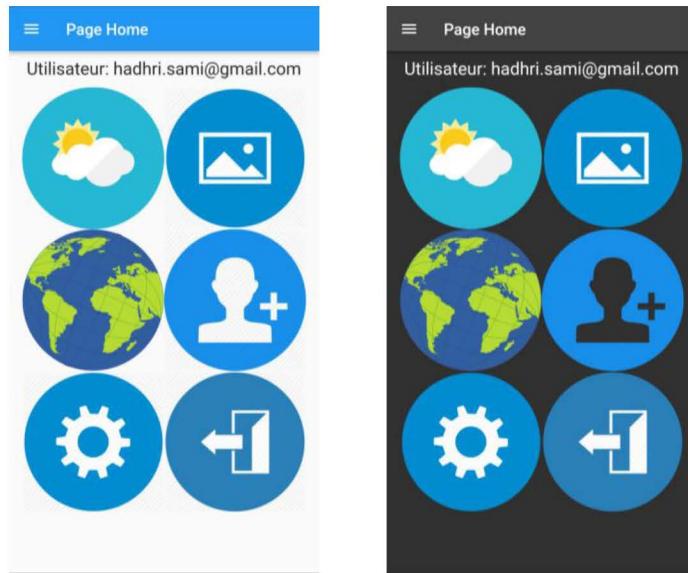
Données Règles Sauvegardes Utilisation

Protégez vos ressources Realtime Database des utilisations abusives telles que la fraude à la facturation et le hameçonnage Configurer App Check X

https://flutter1-59780-default-rtdb.firebaseio.com

mode: "Nuit"

On remarque que la variable mode a changé dans la console mais le changement de mode n'est pas appliqué instantanément : il faudrait plutôt redémarrer l'application.



Afin de palier à ce problème, il faudrait que la variable « mode » du fichier « parametres.page.dart » soit accessible de n'importe quelle page de l'application. C'est pourquoi nous allons utiliser les NotifyListeners de DART : cela permet de s'abonner à un écouteur sur la variable mode et d'informer tous les abonnés à chaque changement de la valeur de cette variable.

14. Créer un dossier « lib/notifier » contenant un fichier « theme.dart ». Implémenter une classe « MonTheme » avec une variable appelée mode définie à "Jour" et 2 méthodes :

- setMode(String m) : pour modifier le mode et informer tous les abonnés
- getTheme() : pour récupérer le Theme

Fichier theme.dart

```
import 'package:flutter/material.dart';

class MonTheme extends ChangeNotifier{
  static String mode="Jour";

  void setMode(String m){
    mode=m;
    notifyListeners();
  }

  ThemeData getTheme(){
    return (mode=="Jour")? ThemeData.light():ThemeData.dark();
  }
}
```

15. Dans le fichier « lib/config/global.params.dart », créer une seule instance de la classe MonTheme appelée themeActuel et la rendre accessible et globale partout dans l'application.

Fichier global.params.dart

```
import 'package:flutter/material.dart';
import '../notifier/theme.dart';

class GlobalParams{
  ...
  static MonTheme themeActuel=MonTheme();
}
```

16. Modifier la méthode _onSaveMode() dans le fichier « parametres.page.dart » pour modifier le mode de la variable globale MonTheme.

Fichier parametres.page.dart

```
...
import '../config/global.params.dart';
...
_onSaveMode() async {
    GlobalParams.themeActuel.setMode(mode);
    await ref.set({"mode": mode});
    print("mode changé");
    Navigator.pop(context);
}
```

17. Converir le Widget MyApp du fichier « main.dart » en StatefulWidget et ajouter la méthode initState() pour s'abonner au notifier.

Fichier main.dart

```
...
import 'config/global.params.dart';
...

Future<void> main() async {
    WidgetsFlutterBinding.ensureInitialized();
    await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);
    await _onGetMode();
    theme = (await _onGetMode() == "Jour") ? ThemeData.light() :
    ThemeData.dark();
    GlobalParams.themeActuel.setMode(await _onGetMode());
    runApp(MyApp());
}

class MyApp extends StatefulWidget {
    @override
    State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
...
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            debugShowCheckedModeBanner: false,
            routes: routes,
            theme: theme,
            theme: GlobalParams.themeActuel.getTheme(),
            home: StreamBuilder<User?>(
                stream: FirebaseAuth.instance.authStateChanges(),
                builder: (context, snapshot) {
                    if (snapshot.hasData)
                        return HomePage();
                    else
                        return AuthenticationPage();
                },
            ),
        );
    }
    //fin de build
}

    @override
    void initState() {
        super.initState();
        GlobalParams.themeActuel.addListener(() {
            setState(() {});
        });
    }
}
...
```

}

☰ Page Paramètres

Mode

Jour

Nuit

Enregistrer

← Page Meteo Details sfax

Sat-29/10/2022 12:00 Clouds	26 °C
Sat-29/10/2022 15:00 Clouds	26 °C
Sat-29/10/2022 18:00 Clear	25 °C
Sat-29/10/2022 21:00 Clear	23 °C
Sun-30/10/2022 00:00 Clear	23 °C
Sun-30/10/2022 03:00 Clear	22 °C
Sun-30/10/2022 06:00 Clear	22 °C

← tunisia, Page 1/24



tunisia, city, tourism



tunisia, flag, national flag

← Page Pays Details palestine



Palestine, State of
فُلَسْطِينُ الْمَلَكَوِيَّةُ

Administration
Capitale : Ramallah
Language(s) : Arabic, العربية

Géographie
Région : Asia
Superficie : null
Fuseau Horaire : UTC+02:00

Démographie
Population : 4803269