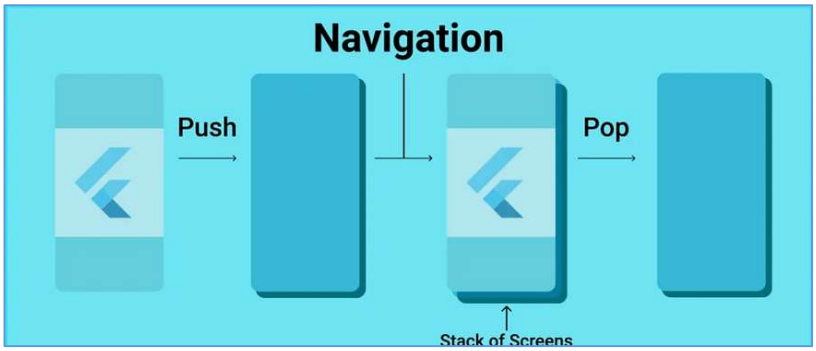
Atelier 4 Flutter
Navigation et Routing



I. Introduction

La navigation dans Flutter est le processus de déplacement d'un écran à un autre. Flutter utilise un système de navigation basé sur une pile, ce qui signifie que lorsque vous naviguez vers un nouvel écran, l'écran précédent est ajouté à une pile. Lorsque vous revenez en arrière, l'écran précédent est supprimé de la pile et affiché à nouveau.

Le flux de travail de toute application mobile est déterminé par la capacité de l'utilisateur à naviguer vers différentes pages ; ce processus est connu sous le nom de routage. La classe **MaterialPageRoute** du système de routage de Flutter est fournie avec les deux méthodes qui montrent comment basculer entre deux itinéraires :

Navigator.push() — Pour naviguer vers l'écran suivant.

Navigator.pop() — Pour revenir à l'écran précédent.

Le widget **Navigator** gère les routes à l'aide d'une approche par pile. Les routes sont empilées en fonction des actions de l'utilisateur, et appuyer sur retour permet de naviguer vers la route la plus récente.

Cependant, si vous devez accéder au même écran dans de nombreuses parties de votre application, cette approche peut entraîner une duplication de code. La solution consiste à définir une route nommée et à l'utiliser pour la navigation.

Pour utiliser des routes nommées, utilisez la fonction **Navigator.pushNamed()**.

II. Mise en place de Navigation et routing

1. Construction de route

lib > main.dart

```dart
// Importation du package Flutter Material
import 'package:flutter/material.dart';
import 'package:myflutterapplication/screens/menu.dart';
import 'package:myflutterapplication/screens/myproducts.dart';
import 'package:myflutterapplication/widgets/myappbar.dart';
import 'package:myflutterapplication/widgets/mybottomnavbar.dart';
import 'package:myflutterapplication/widgets/mydrawer.dart';

// Fonction principale qui lance l'application Flutter
void main() {
  // Lance l'application en exécutant MyApp
  runApp(const MyApp());
}

// Définition de la classe stateless MyApp
class MyApp extends StatelessWidget {
  // Constructeur constant avec une clé facultative
  const MyApp({super.key});

  // Ce widget est la racine de l'application
  @override
  Widget build(BuildContext context) {
    // Retourne un MaterialApp configuré
    return MaterialApp(
      // Titre de l'application
      title: 'Flutter Application',
      // Thème de l'application avec une palette de couleurs personnalisée
      theme: ThemeData(
        // Utilisation d'un ColorScheme basé sur une couleur de départ
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        // Activation de Material Design 3
        useMaterial3: true,
      ),
      // Désactivation du bandeau "DEBUG"
      debugShowCheckedModeBanner: false,
      initialRoute: '/',
      routes: {
        '/Items': (context) => const Myproducts(),
      },

      // Définition de la page d'accueil de l'application
      home: const Scaffold(
        // AppBar de l'application
        appBar: Myappbar(),
        // Corps du Scaffold
        body: Menu(),
        drawer: MyDrawer(),
        bottomNavigationBar: MyBottomNavigation(),
      ),
```

```
    );
  }
}
```

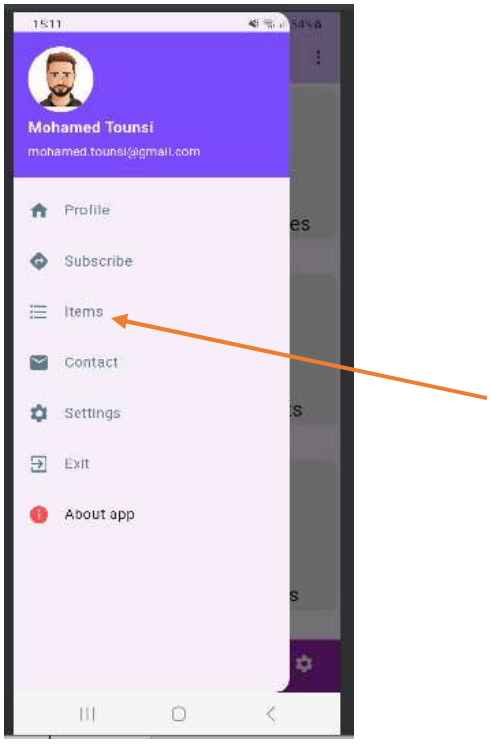
lib > widgets > mydrawer.dart

```
      ….      return ListTile(
              leading: Icon(choice.icon, color: Colors.blueGrey),
              textColor: Colors.blueGrey,
              title: Text(choice.title),
              onTap: () {
                print(choice.title);
                 Navigator.pushNamed(context, '/Items');
              },
            );
….
```

Explication

context est une référence à l'arbre de widgets dans lequel cette méthode est appelée. Il est utilisé pour accéder au Navigator qui est nécessaire pour pousser une nouvelle route.

Résultat



2. Fichier de route séparé

Créer le fichier /lib/**approuter.dart**


```dart
// Importation des packages nécessaires
import 'package:flutter/material.dart';
import 'package:myflutterapplication/screens/menu.dart';
import 'package:myflutterapplication/screens/myproducts.dart';
import 'package:myflutterapplication/widgets/myappbar.dart';
import 'package:myflutterapplication/widgets/mybottomnavbar.dart';
import 'package:myflutterapplication/widgets/mydrawer.dart';

// Définition d'une fonction qui retourne les routes
Map<String, WidgetBuilder> appRoutes() {
  return {
    '/': (context) => const Scaffold(
          appBar: Myappbar(),
          body: Menu(),
          drawer: MyDrawer(),
          bottomNavigationBar: MyBottomNavigation(),
        ),
    '/Items': (context) => Scaffold(
          appBar: AppBar(
            title: const Text('My Products'),
          ),
          body: const Myproducts(),
          drawer: const MyDrawer(),
          bottomNavigationBar: const MyBottomNavigation(),
        ),
```

```
  };
}
```

Explication

Nous avons créé 2 routes / et /items.

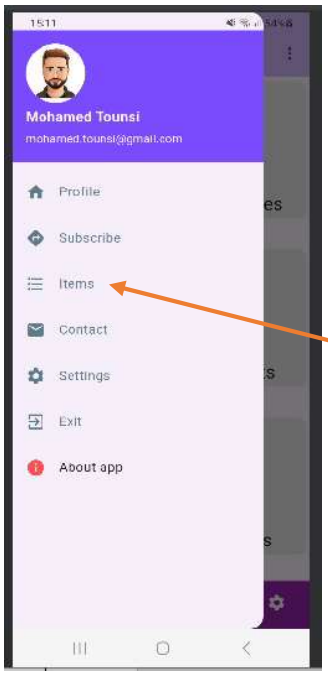
Dans la route / nous y avons **couper** et coller le contenu qui existait dans main.dart concernant **le Scaffold et l'appel de Home.**

Concernant la route /items nous y avons fait appel le screen Myproducts dans body. On y a changé l'AppBar et nous y avons fait appel le même drawer et bottomNavigationBar que Home.


```dart
// Importation du package Flutter Material
import 'package:flutter/material.dart';
import 'package:myflutterapplication/approuter.dart';

// Fonction principale qui lance l'application Flutter
void main() {
  // Lance l'application en exécutant MyApp
  runApp(const MyApp());
}

// Définition de la classe stateless MyApp
class MyApp extends StatelessWidget {
  // Constructeur constant avec une clé facultative
  const MyApp({super.key});

  // Ce widget est la racine de l'application
  @override
  Widget build(BuildContext context) {
    // Retourne un MaterialApp configuré
    return MaterialApp(
      // Titre de l'application
      title: 'Flutter Application',
      // Thème de l'application avec une palette de couleurs personnalisée
      theme: ThemeData(
        // Utilisation d'un ColorScheme basé sur une couleur de départ
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        // Activation de Material Design 3
        useMaterial3: true,
      ),
      // Désactivation du bandeau "DEBUG"
      debugShowCheckedModeBanner: false,
       initialRoute: '/',
       routes: appRoutes(),  // Utilisation des routes depuis le fichier séparé
    );
```

```
  }

}
```

Résultat



3. Routes dynamiques avec chaîne interpolée

L'utilisateur peut choisir parmi plusieurs options (par exemple, un menu ou une liste). Chaque option a un titre, et lorsque l'utilisateur sélectionne une option, il est redirigé vers une page spécifique liée à cette option.

lib > widgets > mydrawer.dart

```
 return ListTile(
              leading: Icon(choice.icon, color: Colors.blueGrey),
              textColor: Colors.blueGrey,
              title: Text(choice.title),
              onTap: () {
                Navigator.pushNamed(context, '/${choice.title}');
              },
            );
```

Explication

'/${choice.title}' est une chaîne interpolée. En Dart, l'interpolation de chaîne permet d'insérer la valeur d'une variable à l'intérieur d'une chaîne de caractères.

Si choice.title est par exemple "Items", alors l'instruction '/${choice.title}' deviendra '/Items'

Attention : Il faut nommer les routes exactement de la même façon que les titles dans le tableau choice dans mydrawer.dart pour qu'il y ait correspondance entre route et title

```
const List<Choice> choices = <Choice>[
  Choice(title: 'Profile', icon: Icons.home),
  Choice(title: 'Subscribe', icon: Icons.directions),
  Choice(title: 'Items', icon: Icons.format_list_bulleted_outlined),
  Choice(title: 'Contact', icon: Icons.email_rounded),
  Choice(title: 'Settings', icon: Icons.settings),
  Choice(title: 'Exit', icon: Icons.exit_to_app),
];
```

4. Exit application

Créer un fichier /lib/**screens**/**exitscreen.dart**

lib > screens > exitscreen.dart

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

class ExitScreen extends StatelessWidget {
  const ExitScreen({super.key});

  @override
  Widget build(BuildContext context) {
     Future.delayed(const Duration(seconds: 2), ()  {
```

```
      SystemNavigator.pop(); // Ferme l'application immédiatement après l'affichage
de ce widget
    });

    return const Scaffold(
      body: Center(child: Text('Exiting...')),
    );
  }
}
```

Explication

Future.delayed(const Duration(seconds: 2), () { est une méthode pour différer l'exécution d'un code jusqu'à ce que la construction actuelle du widget soit terminée, ce qui est utile pour certaines actions qui doivent se produire après l'affichage initial du widget.

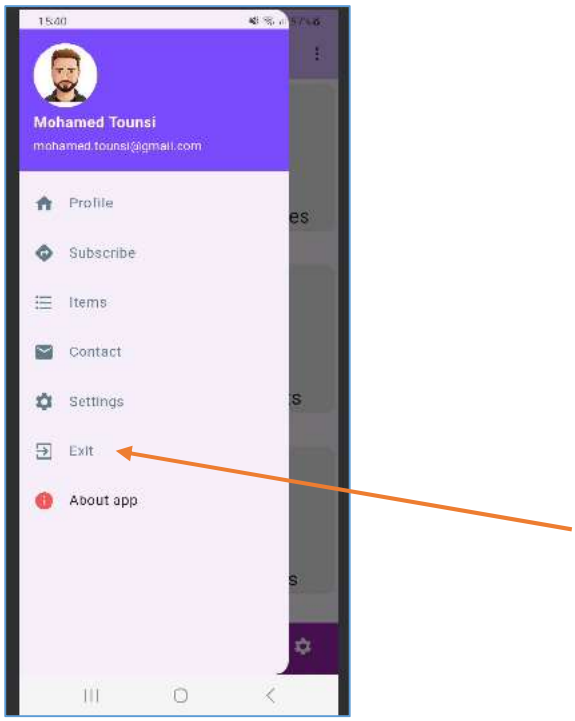
Duration(seconds: 2) attend 2 secondes.

lib > approuter.dart

```
// Importation des packages nécessaires
import 'package:flutter/material.dart';
import 'package:myflutterapplication/screens/exitscreen.dart';
import 'package:myflutterapplication/screens/menu.dart';
import 'package:myflutterapplication/screens/myproducts.dart';
import 'package:myflutterapplication/widgets/myappbar.dart';
import 'package:myflutterapplication/widgets/mybottomnavbar.dart';
import 'package:myflutterapplication/widgets/mydrawer.dart';

// Définition d'une fonction qui retourne les routes
Map<String, WidgetBuilder> appRoutes() {
  return {
    '/': (context) => const Scaffold(
          appBar: Myappbar(),
          body: Menu(),
          drawer: MyDrawer(),
          bottomNavigationBar: MyBottomNavigation(),
        ),
      '/Items': (context) => Scaffold(
          appBar: AppBar(
            title: const Text('My Products'),
          ),
          body: const Myproducts(),
          drawer: const MyDrawer(),
          bottomNavigationBar: const MyBottomNavigation(),
        ),
```

```
        '/Exit': (context) => const ExitScreen(), // Route associée à l'action de
fermeture
  };
}
```

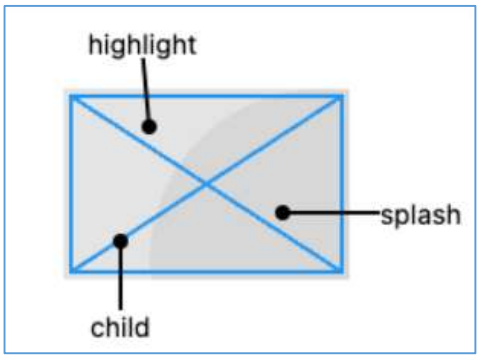
Résultat



III. InkWell

Zone rectangulaire de Material qui réagit au toucher. Il répond à l'action tactile effectuée par l'utilisateur. Inkwell répondra lorsque l'utilisateur cliquera sur le bouton. Il y a tellement de gestes comme appuyer deux fois, appuyer longuement, appuyer vers le bas, etc.

Le widget InkWell doit avoir un widget Material comme ancêtre.

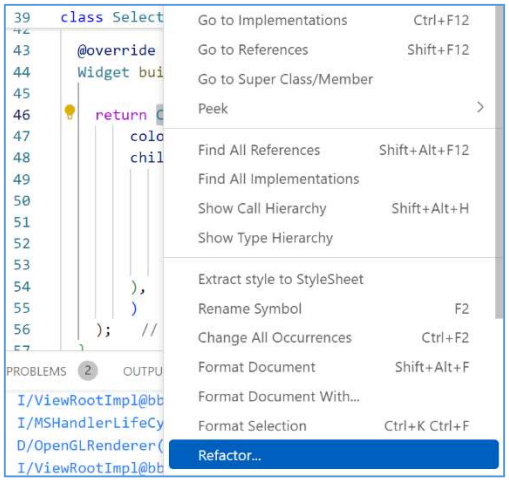


InkWell répond à l'action tactile effectuée par l'utilisateur. Inkwell répondra lorsque l'utilisateur cliquera sur le bouton. Il existe de nombreux gestes comme le double-tap, l'appui long, le tapotement, etc. Vous trouverez ci-dessous les nombreuses propriétés de ce widget. Nous pouvons définir le rayon du widget Inkwell à l'aide de radius et également le rayon de bordure à l'aide de borderRadius. Nous pouvons donner la couleur de l'éclaboussure à l'aide de splashColor et faire beaucoup de choses.

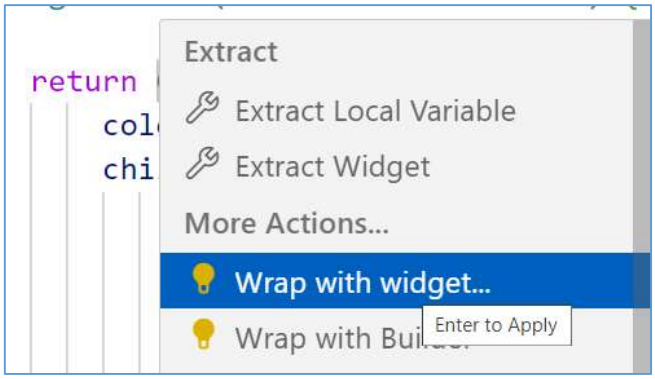
Ouvrir le fichier

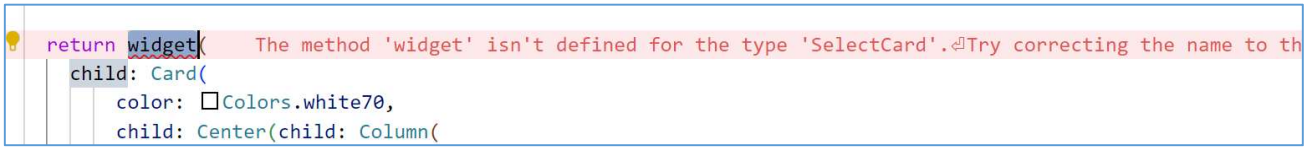
Sélectionner Card puis bouton droit refactor

```
return Card(
    color: Colors.white70,
    child: Center(child: Column(
```



Choisir Wrap with widget

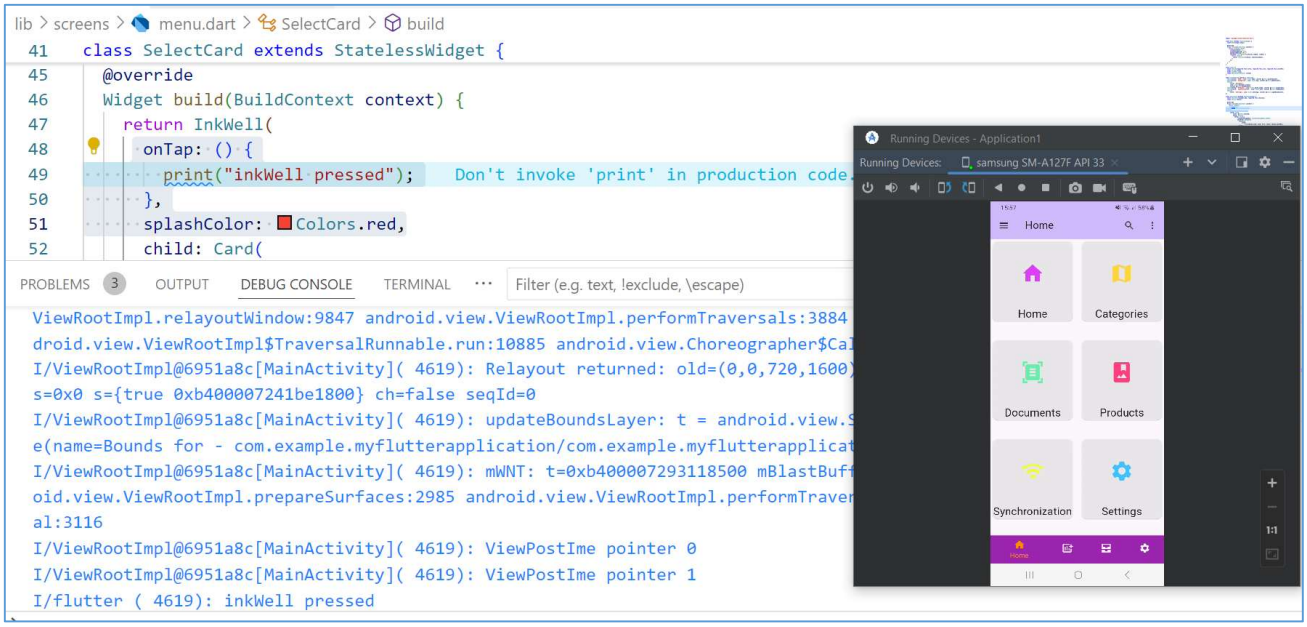




Ecrire InkWell qui remplace widget

Puis y ajouter onTap

```
….
  Widget build(BuildContext context) {
    return InkWell(
      onTap: () {
        print("inkWell pressed");
      },
      splashColor: Colors.red,
      child: Card(
          color: Colors.white70,
…..
```



Créer Fichier /lib/**screens**/**documents.dart**

lib > screens > documents.dart

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


class Documents extends StatelessWidget {
  const Documents({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Documents'),
      ),
      body: const Center(
        child: Text(
          'Welcome to the Documents Screen!',
          style: TextStyle(fontSize: 24),
        ),
      ),
    );
  }
}
```

Préparer la route correspondante.

lib > approuter.dart

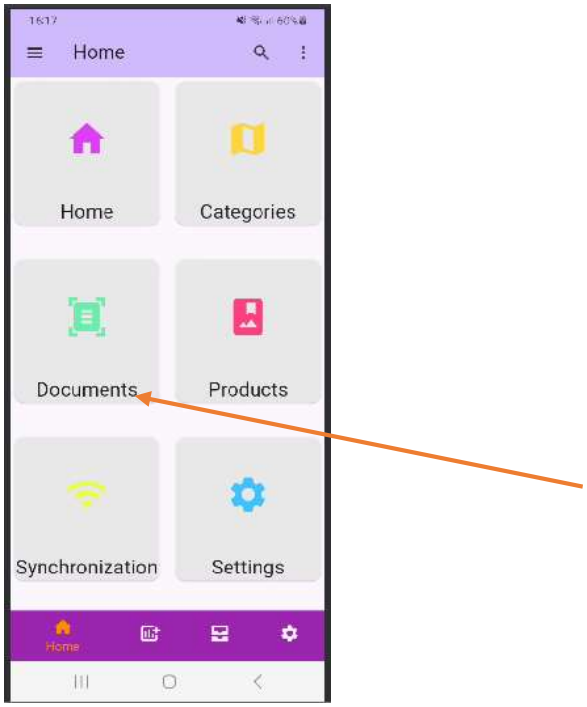
```dart
'/Documents': (context) =>
        const Documents(), // Route pour l'écran Documents
```

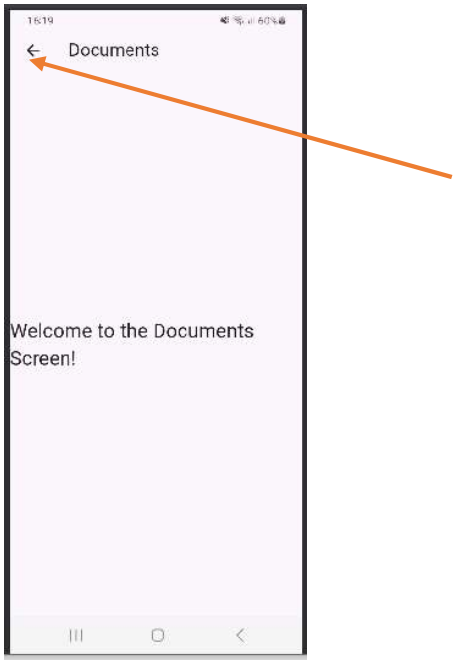
Puis dans onTap on fait appel à la route :

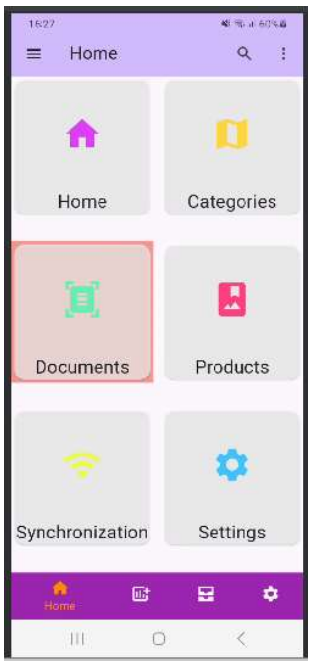
lib > screens > menu.dart

```
.....

 Widget build(BuildContext context) {
    return InkWell(
      onTap: () {
        print("inkWell pressed");
            Navigator.of(context).pushNamed('/${choice.title}');
      },
      splashColor: Colors.red,
      child: Card(
.....
```

Résultat




Documents
Welcome to the Documents Screen!


Home
Home
Categories
Documents
Products
Synchronization
Settings
Home