import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;

import 'dart:convert';

void main() {

  runApp(MaterialApp(

*home*: SubscriptionForm(),

  ));

}

class SubscriptionForm extends StatefulWidget {

  @override

  \_SubscriptionFormState createState() => \_SubscriptionFormState();

}

class \_SubscriptionFormState extends State<SubscriptionForm> {

  final TextEditingController nameController = TextEditingController();

  final TextEditingController postnomController = TextEditingController();

  final TextEditingController prenomController = TextEditingController();

  final TextEditingController genreController = TextEditingController();

  final TextEditingController telephoneController = TextEditingController();

  final TextEditingController tailleController = TextEditingController();

  final TextEditingController poidsController = TextEditingController();

*// final TextEditingController dateInscriptionController = TextEditingController();*

  final TextEditingController adresseController = TextEditingController();

  List<dynamic> abonnes = []; *// Liste pour stocker les informations des abonnés*

  @override

  void initState() {

    super.initState();

    fetchAbonnes(); *// Appel de la fonction pour récupérer les informations des abonnés*

  }

  Future<void> fetchAbonnes() async {

    try {

      var url = Uri.parse("http://localhost:88/APITest/get.php");

      var response = await http.get(url);

      if (response.statusCode == 200) {

        setState(() {

          abonnes = jsonDecode(response.body);

        });

      } else {

        print("Failed to load data: ${*response*.*statusCode*}");

      }

    } catch (ex) {

      print("Erreur lors de la récupération des données : $*ex*");

    }

  }

*// Fonction pour enregistrer les données*

  Future<void> saveData() async {

    try {

      var url = Uri.parse("http://localhost:88/salle\_sport/postabonne.php");

      await http.post(url, *body*: {

        "nom": nameController.text,

        "postnom": postnomController.text,

        "prenom": prenomController.text,

        "genre": genreController.text,

        "telephone": telephoneController.text,

        "taille": tailleController.text,

        "poids": poidsController.text,

        "adresse": adresseController.text,

      });

      print("Données insérées avec succès.");

    } catch (ex) {

      print("Erreur lors de l'insertion : $*ex*");

    }

  }

  bool isAddingSubscription = false; *// Pour contrôler l'affichage des champs de saisie*

*// Fonction pour supprimer un abonné*

Future<void> deleteAbonne(int *idAbonne*) async {

  try {

    var url = Uri.parse("http://localhost:88/salle\_sport/deleteabonne.php?idabonne=$*idAbonne*");

    var response = await http.delete(url);

    if (response.statusCode == 200) {

      setState(() {

*// Appel de fetchAbonnes dans setState pour forcer la mise à jour de l'interface utilisateur*

        fetchAbonnes();

      });

      print("Abonné supprimé avec succès.");

    } else {

      print("Échec de la suppression de l'abonné: ${*response*.*statusCode*}");

    }

  } catch (ex) {

    print("Erreur lors de la suppression de l'abonné : $*ex*");

  }

}

*// Fonction pour afficher une boîte de dialogue de confirmation*

Future<void> confirmDeleteAbonne(int *idAbonne*) async {

*// Convertir l'ID en int si ce n'est pas déjà le cas*

*idAbonne* = int.parse(*idAbonne*.toString());

  return showDialog<void>(

*context*: context,

*barrierDismissible*: false,

*builder*: (BuildContext *context*) {

      return AlertDialog(

*title*: Text('Confirmation'),

*content*: SingleChildScrollView(

*child*: ListBody(

*children*: <Widget>[

              Text('Voulez-vous vraiment supprimer cet abonné ?'),

            ],

          ),

        ),

*actions*: <Widget>[

          TextButton(

*child*: Text('Annuler'),

*onPressed*: () {

              Navigator.of(*context*).pop();

            },

          ),

          TextButton(

*child*: Text('Supprimer'),

*onPressed*: () {

              deleteAbonne(*idAbonne*);

              fetchAbonnes();

              Navigator.of(*context*).pop();

            },

          ),

        ],

      );

    },

  );

}

*// Ajoutez cette méthode à la classe \_SubscriptionFormState*

Future<void> showAbonneUpdateDialog(Map<String, dynamic> *abonneDetails*) async {

  return showDialog<void>(

*context*: context,

*barrierDismissible*: false,

*builder*: (BuildContext *context*) {

      return AlertDialog(

*title*: Text('Modifier les détails de l\'abonné'),

*content*: SingleChildScrollView(

*child*: Column(

*children*: <Widget>[

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['nom']),

*labelText*: 'Nom de l\'abonné',

*prefixIcon*: Icons.person,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['postnom']),

*labelText*: 'Postnom',

*prefixIcon*: Icons.person,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['prenom']),

*labelText*: 'Prénom',

*prefixIcon*: Icons.person,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['genre']),

*labelText*: 'Genre',

*prefixIcon*: Icons.person,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['telephone']),

*labelText*: 'Téléphone',

*prefixIcon*: Icons.phone,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['taille']),

*labelText*: 'Taille',

*prefixIcon*: Icons.height,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['poids']),

*labelText*: 'Poids',

*prefixIcon*: Icons.fitness\_center,

              ),

              SizedBox(*height*: 16),

              \_buildTextField(

*controller*: TextEditingController(*text*: *abonneDetails*['adresse']),

*labelText*: 'Adresse',

*prefixIcon*: Icons.location\_on,

              ),

            ],

          ),

        ),

*actions*: <Widget>[

          TextButton(

*child*: Text('Annuler'),

*onPressed*: () {

              Navigator.of(*context*).pop();

            },

          ),

          TextButton(

*child*: Text('Enregistrer'),

*onPressed*: () {

*// Ajoutez la logique pour enregistrer les modifications ici*

              updateAbonne(*abonneDetails*);

              Navigator.of(*context*).pop(); *// Fermer la boîte de dialogue après la mise à jour*

            },

          ),

        ],

      );

    },

  );

}

*// Ajoutez cette méthode à la classe \_SubscriptionFormState pour mettre à jour les détails de l'abonné*

Future<void> updateAbonne(Map<String, dynamic> *abonneDetails*) async {

  try {

    var url = Uri.parse("http://localhost:88/salle\_sport/updateabonne.php");

    var response = await http.put(url, *body*: {

      "idabonne": *abonneDetails*['idabonne'],

      "nom": nameController.text,

      "postnom": postnomController.text,

      "prenom": prenomController.text,

      "genre": genreController.text,

      "telephone": telephoneController.text,

      "taille": tailleController.text,

      "poids": poidsController.text,

      "adresse": adresseController.text,

    });

    if (response.statusCode == 200) {

*// Mettre à jour l'interface utilisateur après la mise à jour réussie*

      fetchAbonnes();

      print("Mise à jour des détails de l'abonné réussie.");

    } else {

      print("Échec de la mise à jour des détails de l'abonné: ${*response*.*statusCode*}");

    }

  } catch (ex) {

    print("Erreur lors de la mise à jour des détails de l'abonné : $*ex*");

  }

}

  @override

  Widget build(BuildContext *context*) {

    return Scaffold(

*appBar*: AppBar(

*title*: Text('Enregistrement Abonné'),

      ),

*body*: SingleChildScrollView(

*child*: Column(

*children*: [

            \_buildHeader(),

            Padding(

*padding*: const EdgeInsets.all(16.0),

*child*: isAddingSubscription

                  ? \_buildAddSubscriptionForm()

                  : \_buildSearchBar(),

            ),

            \_buildAbonneList(), *// Affichage des informations des abonnés*

          ],

        ),

      ),

*bottomNavigationBar*: \_buildBottomNavigationBar(),

    );

  }

  Widget \_buildHeader() {

    return Container(

*padding*: EdgeInsets.all(16),

*color*: Colors.blue,

*child*: Center(

*child*: Image.asset(

          'assets/logo1.jpeg', *// Assurez-vous de remplacer par le chemin correct*

*height*: 100,

        ),

      ),

    );

  }

  Widget \_buildSearchBar() {

    return Row(

*children*: [

        Expanded(

*child*: Padding(

*padding*: const EdgeInsets.symmetric(*horizontal*: 16),

*child*: TextFormField(

*decoration*: InputDecoration(

*labelText*: 'Rechercher un abonné',

*prefixIcon*: Icon(Icons.search),

*border*: OutlineInputBorder(

*borderRadius*: BorderRadius.circular(8),

                ),

*focusedBorder*: OutlineInputBorder(

*borderSide*: BorderSide(*color*: Colors.blue),

*borderRadius*: BorderRadius.circular(8),

                ),

              ),

            ),

          ),

        ),

        IconButton(

*onPressed*: () {

            setState(() {

              isAddingSubscription = true;

            });

          },

*icon*: Icon(Icons.add),

*color*: Colors.blue,

        ),

      ],

    );

  }

  Widget \_buildAddSubscriptionForm() {

    return Form(

*child*: Column(

*crossAxisAlignment*: CrossAxisAlignment.stretch,

*children*: [

          \_buildTextField(

*controller*: nameController,

*labelText*: 'Nom de l\'abonné',

*prefixIcon*: Icons.person,

          ),

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: postnomController,

*labelText*: 'Postnom',

*prefixIcon*: Icons.person,

          ),

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: prenomController,

*labelText*: 'Prénom',

*prefixIcon*: Icons.person,

          ),

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: genreController,

*labelText*: 'Genre',

*prefixIcon*: Icons.person,

          ),

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: telephoneController,

*labelText*: 'Téléphone',

*prefixIcon*: Icons.phone,

          ),

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: tailleController,

*labelText*: 'Taille',

*prefixIcon*: Icons.height,

          ),

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: poidsController,

*labelText*: 'Poids',

*prefixIcon*: Icons.fitness\_center,

          ),

*// SizedBox(height: 16),*

*// \_buildTextField(*

*//   controller: dateInscriptionController,*

*//   labelText: 'Date d\'inscription',*

*//   prefixIcon: Icons.calendar\_today,*

*// ),*

          SizedBox(*height*: 16),

          \_buildTextField(

*controller*: adresseController,

*labelText*: 'Adresse',

*prefixIcon*: Icons.location\_on,

          ),

          SizedBox(*height*: 16),

          ElevatedButton(

*onPressed*: () {

*// Ajouter la logique pour enregistrer l'abonné*

              saveData();

              setState(() {

                isAddingSubscription = false;

              });

            },

*style*: ElevatedButton.styleFrom(

*primary*: Colors.blue,

*elevation*: 5,

*padding*: EdgeInsets.all(12),

*shape*: RoundedRectangleBorder(

*borderRadius*: BorderRadius.circular(8),

              ),

            ),

*child*: Text('Enregistrer'),

          ),

          SizedBox(*height*: 16),

          IconButton(

*onPressed*: () {

              setState(() {

                isAddingSubscription = false;

              });

              fetchAbonnes();

            },

*icon*: Icon(Icons.close),

*color*: Colors.red,

          ),

        ],

      ),

    );

  }

  Widget \_buildTextField({

    required TextEditingController *controller*,

    required String *labelText*,

    required IconData *prefixIcon*,

  }) {

    return TextFormField(

*controller*: *controller*,

*decoration*: InputDecoration(

*labelText*: *labelText*,

*border*: OutlineInputBorder(

*borderRadius*: BorderRadius.circular(8),

        ),

*focusedBorder*: OutlineInputBorder(

*borderSide*: BorderSide(*color*: Colors.blue),

*borderRadius*: BorderRadius.circular(8),

        ),

*prefixIcon*: Icon(*prefixIcon*),

*filled*: true,

*fillColor*: Colors.grey[200],

      ),

    );

  }

  Widget \_buildAbonneList() {

    return ListView.builder(

*shrinkWrap*: true,

*physics*: NeverScrollableScrollPhysics(),

*itemCount*: abonnes.length,

*itemBuilder*: (*context*, *index*) {

        var abonne = abonnes[*index*];

        return Card(

*margin*: const EdgeInsets.all(10),

*child*: ListTile(

*leading*: const Icon(Icons.person,*color*: const Color.fromARGB(255, 1, 99, 179),*size*: 60,),

*title*: Text('ID: ${*abonne*['idabonne']}'),

*subtitle*: Column(

*crossAxisAlignment*: CrossAxisAlignment.start,

*children*: [

                Text('Noms: ${*abonne*['nom']} ${*abonne*['postnom']} ${*abonne*['prenom']}'),

                Text('Genre: ${*abonne*['genre']}'),

                Text('Telephone: ${*abonne*['telephone']}'),

                Text('Taille: ${*abonne*['taille']}'),

                Text('Poids: ${*abonne*['poids']}'),

*// Text('Date: ${abonne['dateInscription']}'),*

                Text('Adresse: ${*abonne*['adresse']}'),

              ],

            ),

*trailing*: Row(

*mainAxisSize*: MainAxisSize.min,

*children*: [

                        IconButton(

*icon*: const Icon(Icons.delete, *color*: Colors.red),

*onPressed*: () {

                            confirmDeleteAbonne(int.parse(abonne['idabonne']));

                          },

                        ),

                        IconButton(

*icon*: const Icon(Icons.update, *color*: Colors.green),

*onPressed*: () {

                            showAbonneUpdateDialog(abonne);

*// Logique de mise à jour ici*

                          },

                        ),

                        IconButton(

*icon*: const Icon(Icons.print, *color*: Color.fromARGB(255, 3, 7, 3)),

*onPressed*: () {

*// Logique d'impression ici*

                          },

                        ),

                      ],

                    ),

          ),

        );

      },

    );

  }

  Widget \_buildBottomNavigationBar() {

    return BottomAppBar(

*child*: Container(

*padding*: EdgeInsets.symmetric(*vertical*: 8),

*child*: Row(

*mainAxisAlignment*: MainAxisAlignment.center,

*children*: [

            ElevatedButton.icon(

*onPressed*: () {

*// Logique pour imprimer la facture*

              },

*style*: ElevatedButton.styleFrom(

*primary*: Colors.green,

*elevation*: 5,

*padding*: EdgeInsets.all(12),

*shape*: RoundedRectangleBorder(

*borderRadius*: BorderRadius.circular(8),

                ),

              ),

*icon*: Icon(Icons.print), *// Icône à côté du texte*

*label*: Text('Imprimer Facture'),

            ),

          ],

        ),

      ),

    );

  }

}