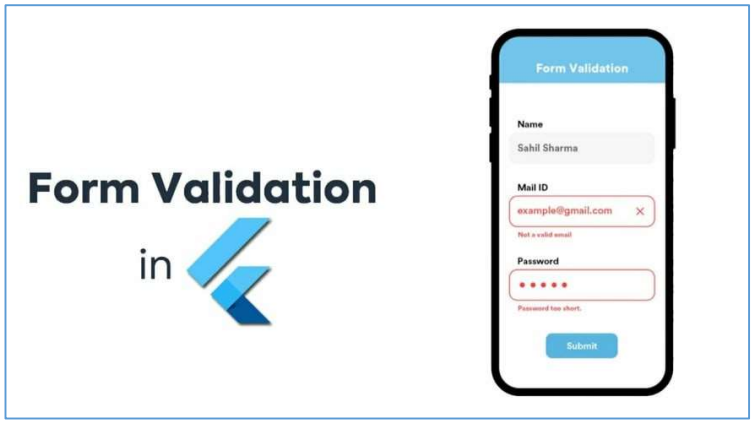
Atelier 6 Flutter
Formulaire et Validation



Les formulaires font partie intégrante de toutes les applications mobiles et Web modernes. Il est principalement utilisé pour interagir avec l'application ainsi que pour recueillir des informations auprès des utilisateurs. Ils peuvent effectuer de nombreuses tâches, qui dépendent de la nature des exigences et de la logique de votre entreprise, telles que l'authentification de l'utilisateur, l'ajout d'un utilisateur, la recherche, le filtrage, la commande, la réservation, etc. Un formulaire peut contenir des champs de texte, des boutons, des cases à cocher, des radios boutons, etc...

I. Form et GlobalKey

Flutter fournit un widget **Form** pour créer un formulaire. Le widget Form agit comme un conteneur, ce qui nous permet de regrouper et de valider les multiples champs du formulaire. Lorsque vous créez un formulaire, il est nécessaire de fournir la **GlobalKey**. Cette clé identifie le formulaire de manière unique et vous permet d'effectuer toute validation dans les champs du formulaire.

```
class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
```

II. TextFormField

Le widget Form utilise le widget enfant **TextFormField** pour permettre aux utilisateurs de saisir le champ de texte. Ce widget affiche un champ de texte de conception de material et nous permet également d'afficher les erreurs de validation lorsqu'elles se produisent.

Nous allons créer une classe. Dans cette classe, nous définissons une clé globale comme _formKey. Cette clé contient un FormState et peut être utilisée pour récupérer le widget Form. Dans la méthode de construction de cette classe, nous allons ajouter un style personnalisé et utiliser le widget TextFormField pour fournir les champs de formulaire tels que le nom, le numéro de téléphone, la date de naissance ou simplement un champ normal. À l'intérieur du TextFormField, nous aurons InputDecoration qui fournit l'apparence des propriétés du formulaire telles que les bordures, les étiquettes, les icônes, les conseils, les styles, etc.

Créer le fichier : /lib/**widgets**/**myform.dart**


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

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
          ),

        ],
      ),
    );
  }
}
```

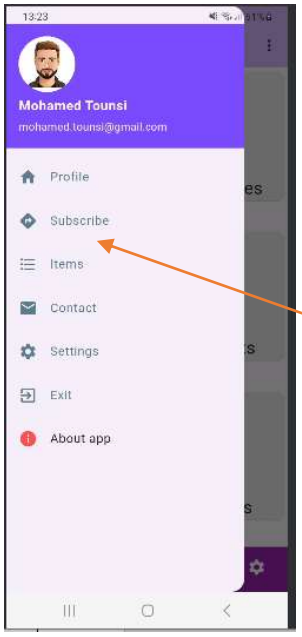
Créer le fichier : /lib/**screens**/**subscribe.dart**

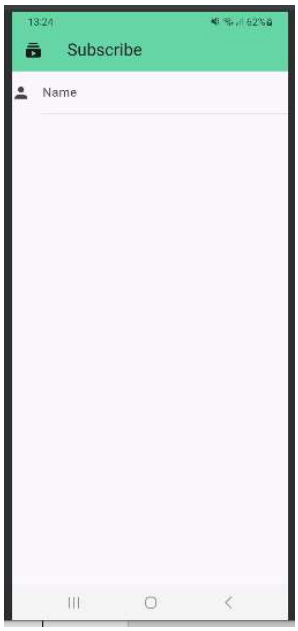

```dart
import 'package:flutter/material.dart';

import '../widgets/myform.dart';

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

  @override
  Widget build(BuildContext context) {
      return Scaffold(
        appBar: AppBar(
          elevation: 15,
          backgroundColor: Colors.greenAccent,
          title: const Text("Subscribe"),
          leading: IconButton(
            onPressed: () {},
            icon: const Icon(Icons.subscriptions_rounded),
          ),
            ),
        body: const Myform(),
    );
  }
}
```

Ajouter la route vers le screen Subscribe

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

Résultat


13:23
Mohamed Tounsi
mohamed.tounsi@gmail.com
Profile
Subscribe
Items
Contact
Settings
Exit
About app


13:24
Subscribe
Name


13:24
Subscribe
Name
Enter your Name

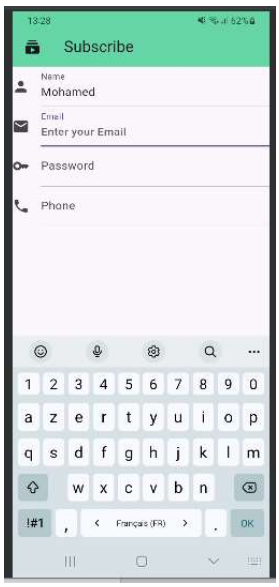
On va ajouter les autres champs de saisi du formulaire.

lib > widgets > myform.dart

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

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
          ),
          TextFormField(
```

```
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
          ),
        ],
      ),
    );
  }
}
```

Résultat



III. ElevatedButton

Nous allons ajouter un bouton pour soumettre le formulaire. Le bouton surélevé est un composant flutter inclus dans le package material, c'est-à-dire « package:flutter/material.dart ». La principale caractéristique de ces boutons est la légère élévation de leur surface vers l'écran lorsque l'utilisateur appuie dessus. Pour gérer le style du bouton surélevé, une classe ButtonStyle est utilisée qui permet de styliser un bouton en fonction des besoins.

lib > widgets > myform.dart

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

```dart
class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
          ),
          Center(
            child: ElevatedButton(
              onPressed: () {},
              style: ButtonStyle(
                backgroundColor: WidgetStateProperty.all<Color>(
                  Colors.greenAccent,
                ),
              ),
              child: const Text("submit"),
            ),
          ),
```

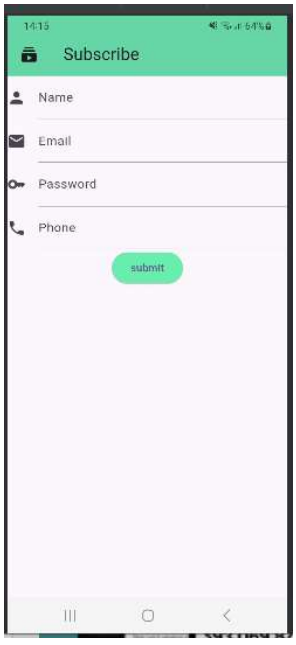
```
        ],
      ),
    );
  }
}
```

Explication

Les propriétés d'état des widgets représentent des valeurs qui dépendent de l'« état » d'un widget. L'état est codé sous la forme d'un ensemble de valeurs WidgetState, comme WidgetState.focused, WidgetState.hovered, WidgetState.pressed.

.all Méthode pratique pour créer une WidgetStateProperty qui se résout en une valeur unique pour tous les états.

Résultat



IV. SnackBar

Pour que votre application soit sûre et facile à utiliser, vous devez vérifier si les informations fournies par l'utilisateur sont valides. Si l'utilisateur a correctement rempli le formulaire, traitez les informations. Si l'utilisateur soumet des informations incorrectes, affichez un message d'erreur convivial pour lui faire savoir ce qui ne s'est pas passé. C'est à ce moment qu'intervient la SnackBar.

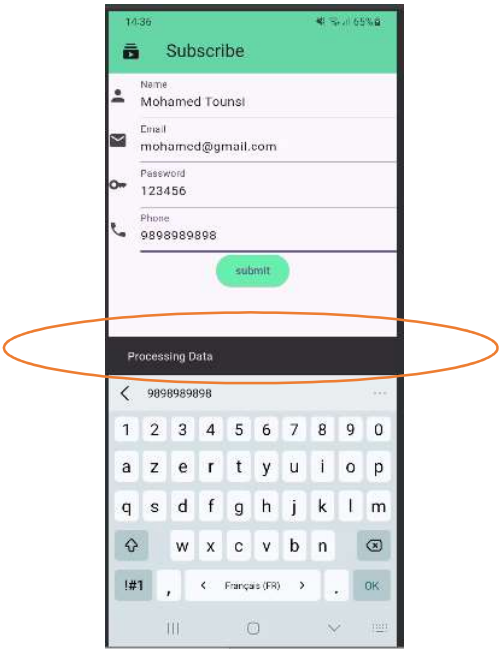
La SnackBar permet d'informer brièvement vos utilisateurs lorsque certaines actions ont lieu. Vous pourriez même leur donner une option pour annuler l'action.

lib > widgets > myform.dart

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

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
          ),
          Center(
            child: ElevatedButton(
              onPressed: () {
                // Retourne true si le formulaire est valide, sinon false
                if (_formKey.currentState!.validate()) {
```

```
                    // Affiche le Snackbar si le formulaire est valide
                    ScaffoldMessenger.of(context).showSnackBar(
                      const SnackBar(content: Text('Processing Data')),
                    );
                  }
                },

                style: ButtonStyle(
                  backgroundColor: WidgetStateProperty.all<Color>(
                    Colors.greenAccent,
                  ),
                ),
                child: const Text("submit"),
              ),
            ),
          ],
        ),
      );
  }
}
```

Résultat



V. Validation du formulaire

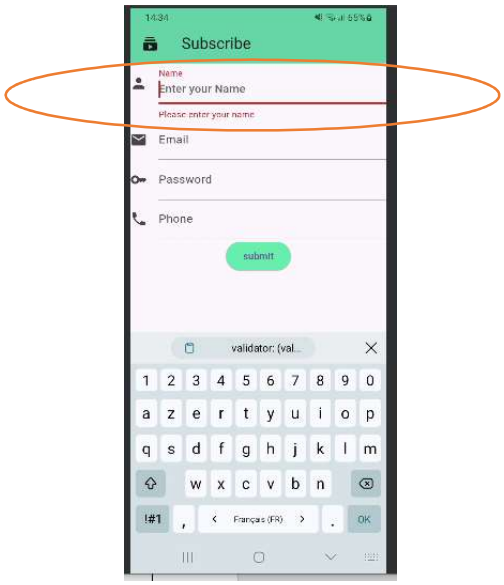
Avec l'option `validator` dans chaque champ.

lib > widgets > myform.dart

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

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your name';
              }
              return null;
            },
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
          ),
          Center(
            child: ElevatedButton(
              onPressed: () {
                // Retourne true si le formulaire est valide, sinon false
```

```
                if (_formKey.currentState!.validate()) {
                  // Affiche le Snackbar si le formulaire est valide
                  ScaffoldMessenger.of(context).showSnackBar(
                    const SnackBar(content: Text('Processing Data')),
                  );
                }
              },
              style: ButtonStyle(
                backgroundColor: WidgetStateProperty.all<Color>(
                  Colors.greenAccent,
                ),
              ),
              child: const Text("submit"),
            ),
          ),
        ],
      ),
    );
  }
}
```

Résultat



On peut mettre par la suite la validation au niveau de chaque champ.

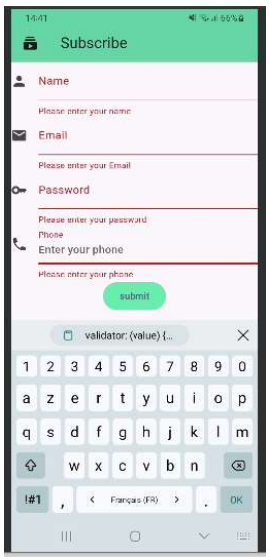
lib > widgets > myform.dart

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

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
```

```
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your name';
              }
              return null;
            },
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
                 validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your Email';
              }
              return null;
            },
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
                 validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your password';
              }
              return null;
            },
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
```

```
                validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your phone';
              }
              return null;
            },
          ),
          Center(
            child: ElevatedButton(
              onPressed: () {
                // Retourne true si le formulaire est valide, sinon false
                if (_formKey.currentState!.validate()) {
                  // Affiche le Snackbar si le formulaire est valide
                  ScaffoldMessenger.of(context).showSnackBar(
                    const SnackBar(content: Text('Processing Data')),
                  );
                }
              },
              style: ButtonStyle(
                backgroundColor: WidgetStateProperty.all<Color>(
                  Colors.greenAccent,
                ),
              ),
              child: const Text("submit"),
            ),
          ),
        ],
      ),
    );
  }
}
```

Résultat



VI. DropdownButton

DropdownButton est un bouton de menu déroulant que nous pouvons utiliser pour sélectionner une valeur parmi un ensemble de valeurs. Dans l'état par défaut, un bouton déroulant affiche sa valeur actuellement sélectionnée. Après avoir cliqué sur le bouton déroulant, il affiche un menu déroulant avec toutes les autres valeurs disponibles, parmi lesquelles l'utilisateur peut en sélectionner une nouvelle.

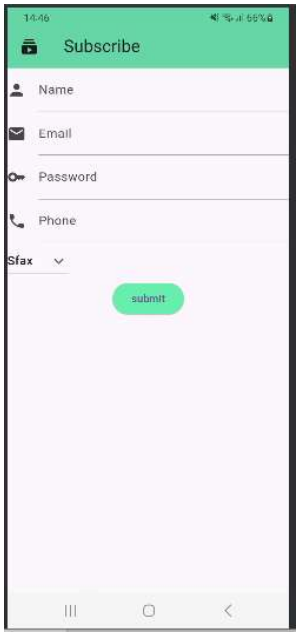
Un bouton déroulant Flutter est un widget de type générique, ce qui signifie que vous pouvez transmettre n'importe quel type de données selon vos besoins. Dans la liste déroulante du menu déroulant, les chaînes de caractères sont utilisées la plupart du temps.

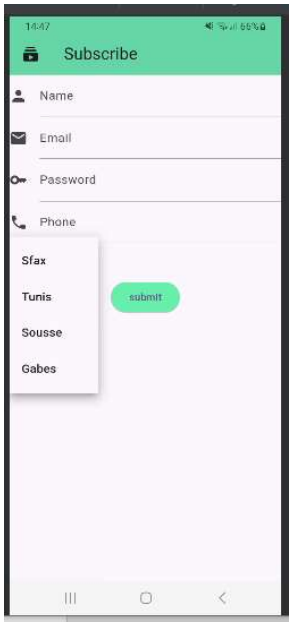
```dart
import 'package:flutter/material.dart';

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
     String? ville = 'Sfax';

     var items = [
    'Sfax',
    'Tunis',
    'Sousse',
    'Gabes'
  ];
  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your name';
              }
              return null;
            },
          ),
```

```
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
                 validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your Email';
              }
              return null;
            },
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
                 validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your password';
              }
              return null;
            },
          ),
          TextFormField(
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
                 validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your phone';
              }
              return null;
            },
          ),
        DropdownButton(
              value: ville,
              icon: const Icon(Icons.keyboard_arrow_down),
               items: items.map((items) {
                return DropdownMenuItem(value: items, child: Text(items));
              }).toList(),
              onChanged: (String? newValue) {
                setState(() {
                  ville = newValue;
                });
              },
            ),
          Center(
            child: ElevatedButton(
              onPressed: () {
                // Retourne true si le formulaire est valide, sinon false
```

```
                if (_formKey.currentState!.validate()) {
                  // Affiche le Snackbar si le formulaire est valide
                  ScaffoldMessenger.of(context).showSnackBar(
                    const SnackBar(content: Text('Processing Data')),
                  );
                }
              },
              style: ButtonStyle(
                backgroundColor: WidgetStateProperty.all<Color>(
                  Colors.greenAccent,
                ),
              ),
              child: const Text("submit"),
            ),
          ),
        ],
      ),
    );
  }
}
```

Résultat





VII. TextEditingController

Un contrôleur pour un champ de texte modifiable.

Chaque fois que l'utilisateur modifie un champ de texte avec un TextEditingController associé, le champ de texte met à jour la valeur et le contrôleur notifie ses écouteurs. Les auditeurs peuvent ensuite lire le texte et les propriétés de la sélection pour savoir ce que l'utilisateur a tapé ou comment la sélection a été mise à jour.

De même, si vous modifiez les propriétés de texte ou de sélection, le champ de texte sera notifié et se mettra à jour de manière appropriée.

Un TextEditingController peut également être utilisé pour fournir une valeur initiale pour un champ de texte. Si vous créez un champ de texte avec un contrôleur qui contient déjà du texte, le champ de texte utilisera ce texte comme valeur initiale.

La valeur (ainsi que le texte et la sélection) de ce contrôleur peut être mise à jour à partir d'un écouteur ajouté à ce contrôleur. Soyez conscient des boucles infinies car l'auditeur sera également informé des modifications apportées à partir de lui-même.

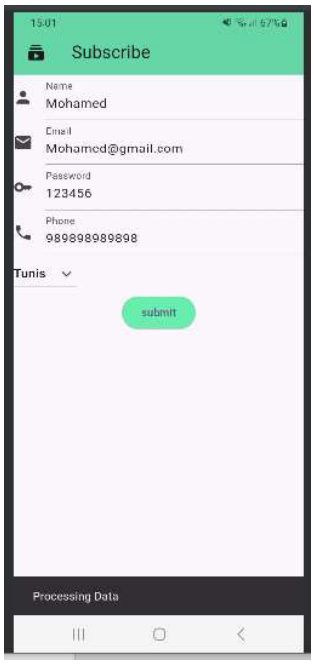
```dart
import 'package:flutter/material.dart';

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
```

```dart
final _formKey = GlobalKey<FormState>();
String? ville = 'Sfax';

var items = ['Sfax', 'Tunis', 'Sousse', 'Gabes'];

late TextEditingController _nameController;
late TextEditingController _emailController;
late TextEditingController _passwordController;
late TextEditingController _phoneController;

@override
void initState() {
  super.initState();
  _nameController = TextEditingController();
  _emailController = TextEditingController();
  _passwordController = TextEditingController();
  _phoneController = TextEditingController();
}

@override
Widget build(BuildContext context) {
  return Form(
    key: _formKey,
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: <Widget>[
        TextFormField(
          controller: _nameController,
          decoration: const InputDecoration(
              icon: Icon(Icons.person),
              hintText: "Enter your Name",
              labelText: "Name"),
          validator: (value) {
            if (value!.isEmpty) {
              return 'Please enter your name';
            }
            return null;
          },
        ),
        TextFormField(
          controller: _emailController,
          decoration: const InputDecoration(
              icon: Icon(Icons.email),
              hintText: "Enter your Email",
              labelText: "Email"),
          validator: (value) {
            if (value!.isEmpty) {
              return 'Please enter your Email';
            }
            return null;
          },
        ),
```

```
          TextFormField(
            controller: _passwordController,
            decoration: const InputDecoration(
                icon: Icon(Icons.key),
                hintText: "Enter your password",
                labelText: "Password"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your password';
              }
              return null;
            },
          ),
          TextFormField(
            controller: _phoneController,
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your phone';
              }
              return null;
            },
          ),
          DropdownButton(
            value: ville,
            icon: const Icon(Icons.keyboard_arrow_down),
            items: items.map((items) {
              return DropdownMenuItem(value: items, child: Text(items));
            }).toList(),
            onChanged: (String? newValue) {
              setState(() {
                ville = newValue;
              });
            },
          ),
          Center(
            child: ElevatedButton(
              onPressed: () {
                // Retourne true si le formulaire est valide, sinon false
                if (_formKey.currentState!.validate()) {
                  print(_nameController.text);
                  print(_emailController.text);
                  print(_passwordController.text);
                  print(_phoneController.text);
                  print(ville);

                  // Affiche le Snackbar si le formulaire est valide
                  ScaffoldMessenger.of(context).showSnackBar(
                    const SnackBar(content: Text('Processing Data')),
```

```
                    );
                  }
                },
                style: ButtonStyle(
                  backgroundColor: WidgetStateProperty.all<Color>(
                    Colors.greenAccent,
                  ),
                ),
                child: const Text("submit"),
              ),
            ),
          ],
        ),
      );
  }
}
```

Résultat




```
I/flutter (21380): Mohamed
I/flutter (21380): Mohamed@gmail.com
I/flutter (21380): 123456
I/flutter (21380): 989898989898
I/flutter (21380): Tunis
```

VIII. obscureText

Pour masquer le mot de passe dans le formulaire.

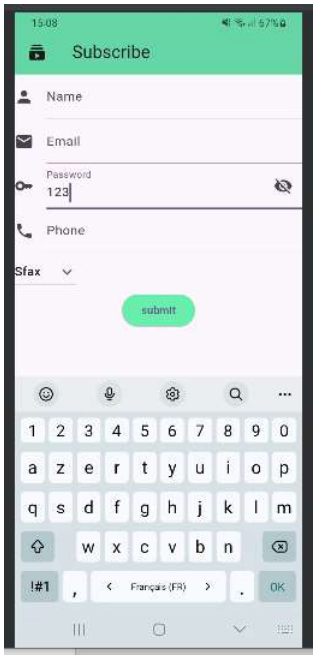
```dart
import 'package:flutter/material.dart';

class Myform extends StatefulWidget {
  const Myform({super.key});

  @override
  State<Myform> createState() => _MyformState();
}

class _MyformState extends State<Myform> {
  final _formKey = GlobalKey<FormState>();
  String? ville = 'Sfax';

  var items = ['Sfax', 'Tunis', 'Sousse', 'Gabes'];

  late TextEditingController _nameController;
  late TextEditingController _emailController;
  late TextEditingController _passwordController;
  late TextEditingController _phoneController;

  @override
  void initState() {
    super.initState();
    _nameController = TextEditingController();
    _emailController = TextEditingController();
    _passwordController = TextEditingController();
    _phoneController = TextEditingController();
  }

 // show the password or not
  bool _isObscure = true;

  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          TextFormField(
            controller: _nameController,
            decoration: const InputDecoration(
                icon: Icon(Icons.person),
                hintText: "Enter your Name",
                labelText: "Name"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your name';
              }
              return null;
```

```
            },
          ),
          TextFormField(
            controller: _emailController,
            decoration: const InputDecoration(
                icon: Icon(Icons.email),
                hintText: "Enter your Email",
                labelText: "Email"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your Email';
              }
              return null;
            },
          ),
          TextFormField(
            obscureText: _isObscure,
            controller: _passwordController,
                decoration: InputDecoration(
                    icon: const Icon(Icons.key),
                    hintText: "Enter your password",
                    labelText: "Password",
                    // this button is used to toggle the password visibility
                suffixIcon: IconButton(
                    icon: Icon(
                        _isObscure ? Icons.visibility : Icons.visibility_off),
                    onPressed: () {
                      setState(() {
                        _isObscure = !_isObscure;
                      }
                      );
                    }
                    ),
                ),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your password';
              }
              return null;
            },
          ),
          TextFormField(
            controller: _phoneController,
            decoration: const InputDecoration(
                icon: Icon(Icons.phone),
                hintText: "Enter your phone",
                labelText: "Phone"),
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please enter your phone';
              }
              return null;
```

```
            },
          ),
          DropdownButton(
            value: ville,
            icon: const Icon(Icons.keyboard_arrow_down),
            items: items.map((items) {
              return DropdownMenuItem(value: items, child: Text(items));
            }).toList(),
            onChanged: (String? newValue) {
              setState(() {
                ville = newValue;
              });
            },
          ),
          Center(
            child: ElevatedButton(
              onPressed: () {
                // Retourne true si le formulaire est valide, sinon false
                if (_formKey.currentState!.validate()) {
                  print(_nameController.text);
                  print(_emailController.text);
                  print(_passwordController.text);
                  print(_phoneController.text);
                  print(ville);

                  // Affiche le Snackbar si le formulaire est valide
                  ScaffoldMessenger.of(context).showSnackBar(
                    const SnackBar(content: Text('Processing Data')),
                  );
                }
              },
              style: ButtonStyle(
                backgroundColor: WidgetStateProperty.all<Color>(
                  Colors.greenAccent,
                ),
              ),
              child: const Text("submit"),
            ),
          ),
        ],
      ),
    );
  }
}
```

Résultat


15:08
Subscribe
Name
Email
Password
Phone
Sfax
submit
Français (FR)
OK


15:08
Subscribe
Name
Email
Password
123
Phone
Sfax
submit
Français (FR)
OK