***Exp 4***

**Aim:** To create an interactive Form using form widget

Theory:

A **Form** in Flutter is a container for user input fields, providing validation and easy state management. It helps in collecting and processing user data like **name, email, password, currency, etc.**

**Key Components of Forms in Flutter**

1. **Form Widget**
   * A wrapper that manages multiple input fields (TextFormField, DropdownButtonFormField, etc.).
   * Uses a GlobalKey<FormState> to validate and save data.
2. **TextFormField**
   * Used for text input like **name, email, password**.
   * Supports built-in validation.
3. **DropdownButtonFormField**
   * Used for selecting values like **currency, country, category**.
4. **CheckboxListTile / Switch**
   * Used for **boolean inputs** (e.g., accept terms & conditions).
5. **ElevatedButton (Submit Button)**
   * Triggers form validation and submission.

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

debugShowCheckedModeBanner: false,

home: CurrencyFormScreen(),

);

}

}

class CurrencyFormScreen extends StatefulWidget {

@override

\_CurrencyFormScreenState createState() => \_CurrencyFormScreenState();

}

class \_CurrencyFormScreenState extends State<CurrencyFormScreen> {

final \_formKey = GlobalKey<FormState>();

String \_name = '';

String? \_selectedCurrency;

final List<String> \_currencies = [

'USD - US Dollar',

'EUR - Euro',

'INR - Indian Rupee',

'GBP - British Pound',

'JPY - Japanese Yen',

];

void \_submitForm() {

if (\_formKey.currentState!.validate()) {

\_formKey.currentState!.save();

ScaffoldMessenger.of(context).showSnackBar(

SnackBar(content: Text('Name: $\_name\nCurrency: $\_selectedCurrency')),

);

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: Text('Currency Form')),

body: Padding(

padding: EdgeInsets.all(16.0),

child: Form(

key: \_formKey,

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

// Name Input Field

TextFormField(

decoration: InputDecoration(labelText: 'Full Name'),

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your name';

}

return null;

},

onSaved: (value) {

\_name = value!;

},

),

SizedBox(height: 16),

// Currency Dropdown

DropdownButtonFormField<String>(

decoration: InputDecoration(labelText: 'Select Currency'),

items: \_currencies.map((currency) {

return DropdownMenuItem(

value: currency,

child: Text(currency),

);

}).toList(),

onChanged: (value) {

setState(() {

\_selectedCurrency = value;

});

},

validator: (value) =>

value == null ? 'Please select a currency' : null,

),

SizedBox(height: 24),

// Submit Button

ElevatedButton(

onPressed: \_submitForm,

child: Text('Submit'),

),

],

),

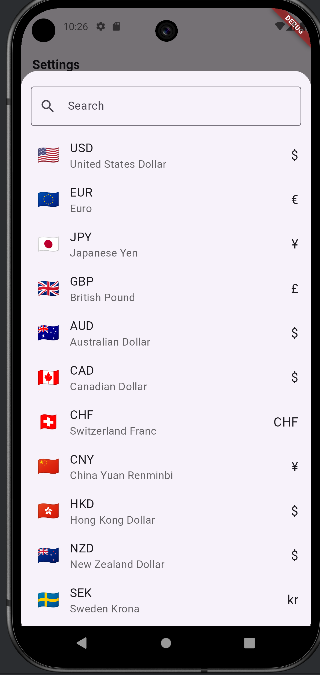
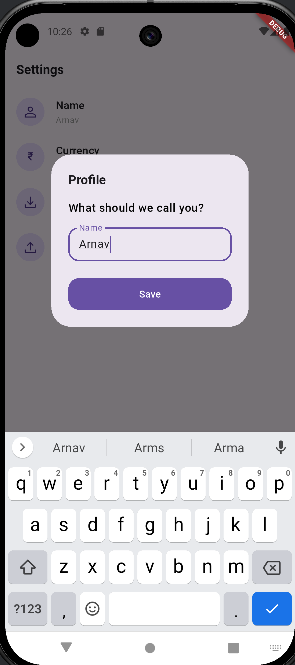
),

),

);

}

}



C

Conclusion:

Forms in Flutter provide a structured way to collect and validate user input efficiently. By using widgets like Form, TextFormField, and DropdownButtonFormField, developers can create interactive and user-friendly input fields. With GlobalKey<FormState>, forms can be validated and processed easily.