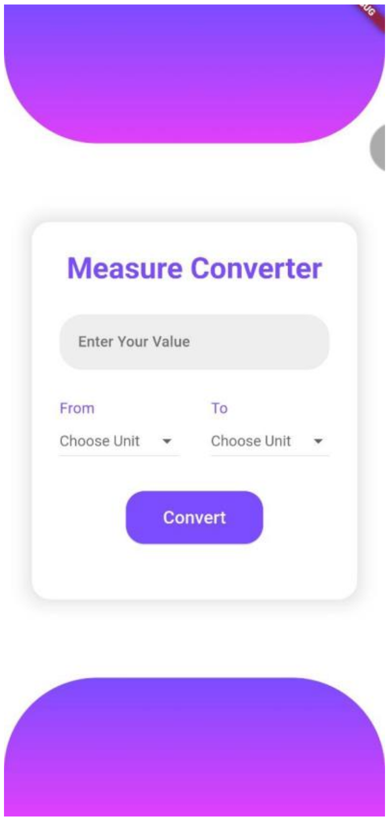


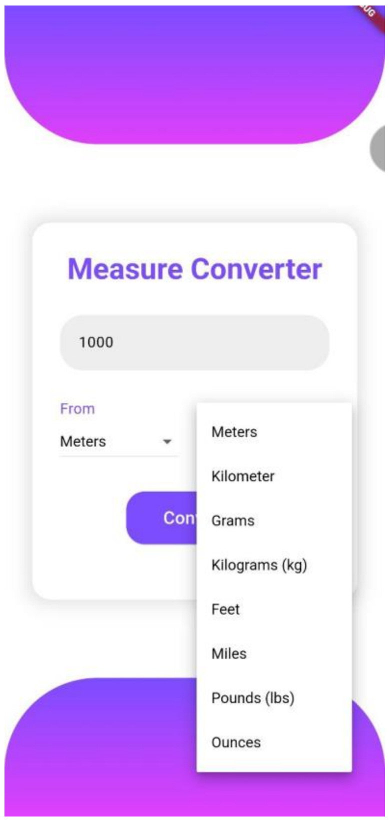
Measure Converter

Enter Your Value

From Choose Unit

To Choose Unit

Convert

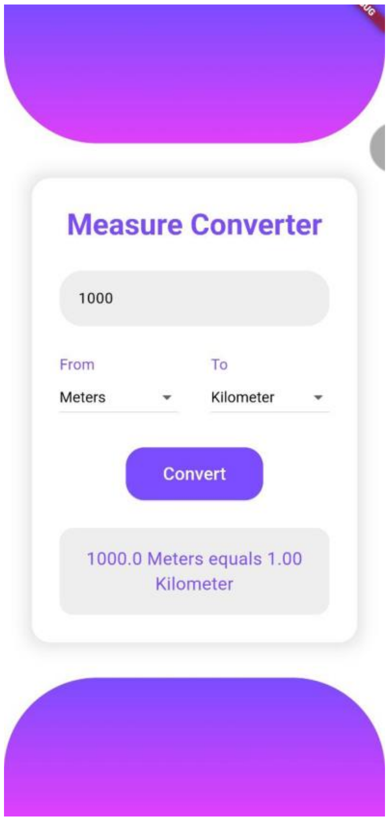


Measure Converter

1000

From Meters

Meters
Kilometer
Grams
Kilograms (kg)
Feet
Miles
Pounds (lbs)
Ounces



Measure Converter

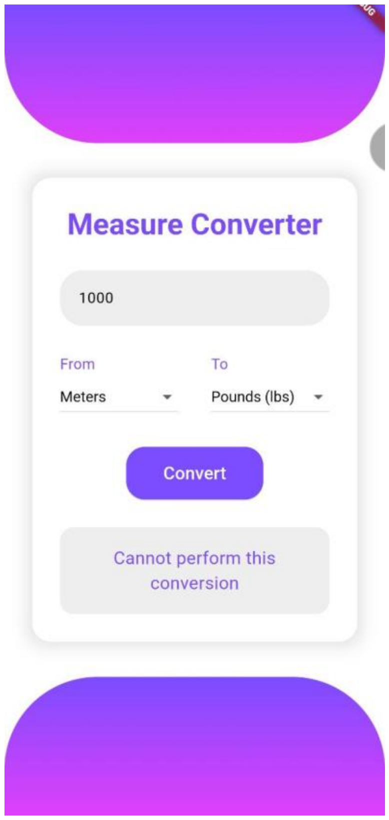
1000

From Meters

To Kilometer

Convert

1000.0 Meters equals 1.00 Kilometer



Measure Converter

1000

From Meters

To Pounds (lbs)

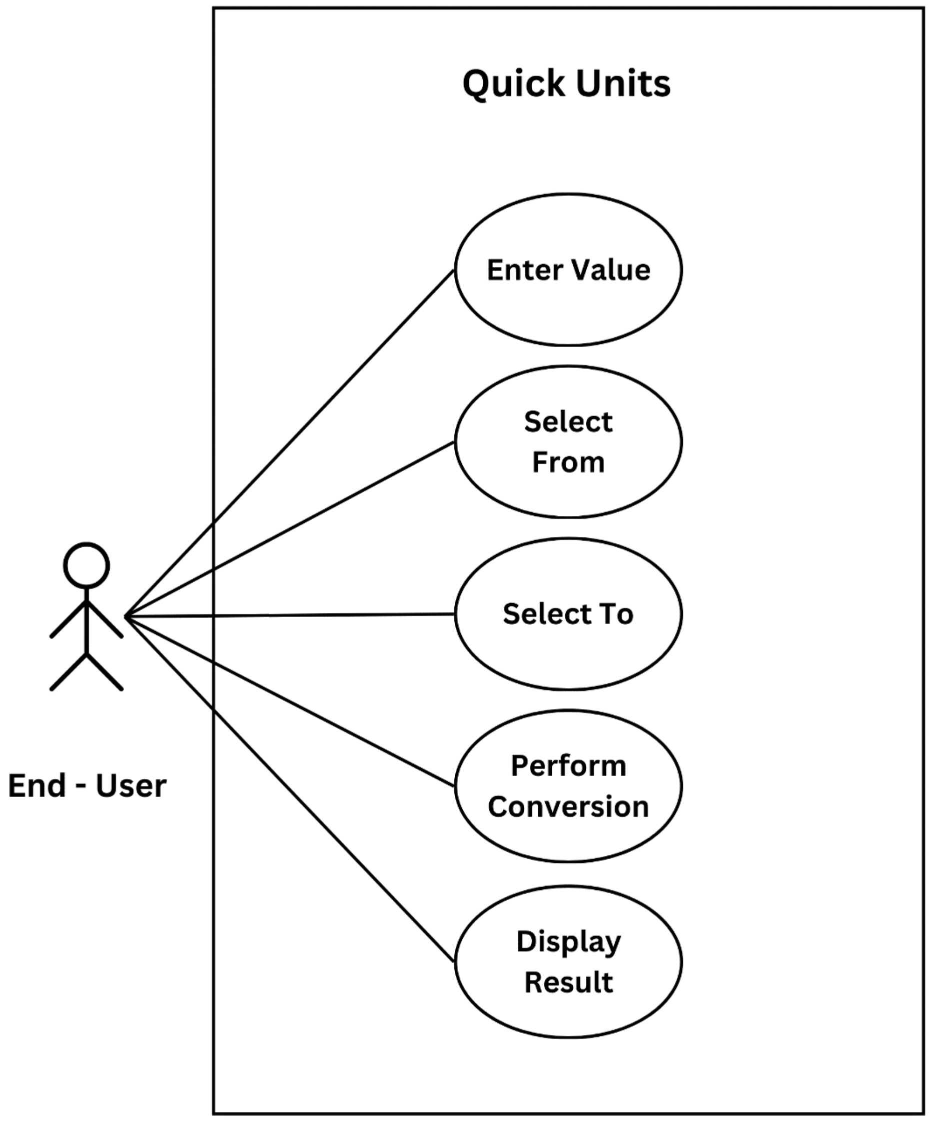
Convert

Cannot perform this conversion

Quick Units

This Flutter app is a tool for converting measurements between different units. It has a simple and easy-to-use interface with these main features:

- **User Input:** You can type in the number you want to convert in a text field.
- **Unit Selection:** There are dropdown menus where you can choose the units you're converting from (like meters or kilograms) and the units you're converting to (like feet or pounds).
- **Conversion Logic:** The app has built-in formulas that do the math for you. If the conversion works, it shows you the result. If it doesn't (like trying to convert meters to grams), it gives you an error message.
- **User Interface:** The app looks nice with a clean design, using deep purple colors, rounded corners, and cool gradient effects at the top and bottom.
- **Error Handling:** If something goes wrong with the conversion, the app will let you know with an error message.
- **Responsive Design:** The app adjusts itself to look good and work smoothly on any screen size.


Quick Units
Enter Value
Select From
Select To
Perform Conversion
Display Result
End - User

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

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      theme: ThemeData(
        primaryColor: Colors.deepPurpleAccent,
        scaffoldBackgroundColor: Colors.white,
        textTheme: TextTheme(
          headlineLarge: TextStyle(
            fontSize: 40,
            fontWeight: FontWeight.bold,
            color: Colors.deepPurpleAccent,
          ),
          labelLarge: TextStyle(fontSize: 24, color: Colors.white),
        ),
        elevatedButtonTheme: ElevatedButtonThemeData(
          style: ButtonStyle(
            backgroundColor: MaterialStateProperty.all(Colors.deepPurpleAccent),
            padding: MaterialStateProperty.all(
              EdgeInsets.symmetric(vertical: 15, horizontal: 40),
            ),
            shape: MaterialStateProperty.all(
              RoundedRectangleBorder(
                borderRadius: BorderRadius.circular(20),
              ),
            ),
          ),
        ),
        inputDecorationTheme: InputDecorationTheme(
          filled: true,
          fillColor: Colors.grey[200],
          contentPadding: EdgeInsets.symmetric(horizontal: 20, vertical: 18),
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(25),
            borderSide: BorderSide.none,
          ),
          hintStyle: TextStyle(color: Colors.black54),
        ),
      ),
      home: MeasureConverterApp(),
    );
  }
}

class MeasureConverterApp extends StatefulWidget {
  @override
  _MeasureConverterAppState createState() => _MeasureConverterAppState();
}

class _MeasureConverterAppState extends State<MeasureConverterApp> {
  double _userInput = 0.0;
  String? _convertedMeasure;
  String errorMessage = '';
  String? _startValue;

  final fromUnits = [
    'Meters', 'Kilometer', 'Grams', 'Kilograms (kg)',
    'Feet', 'Miles', 'Pounds (lbs)', 'Ounces'
  ];

  final Map<String, int> measuresMap = {
    'Meters': 0, 'Kilometer': 1, 'Grams': 2, 'Kilograms (kg)': 3,
    'Feet': 4, 'Miles': 5, 'Pounds (lbs)': 6, 'Ounces': 7
  };

  final dynamic formulas = {
    '0': [1, 0.001, 0, 0, 3.28084, 0.000621371, 0, 0],
    '1': [1000, 1, 0, 0, 3280.84, 0.621371, 0, 0],
    '2': [0, 0, 1, 0.001, 0, 0, 0.00220462, 0.03527396],
    '3': [0, 0, 1000, 1, 0, 0, 2.20462, 35.27396],
    '4': [0.3048, 0.0003048, 0, 0, 1, 0.000189394, 0, 0],
    '5': [1609.34, 1.60934, 0, 0, 5280, 1, 0, 0],
    '6': [0, 0, 453.592, 0.453592, 0, 0, 1, 16],
    '7': [0, 0, 28.3495, 0.0283495, 0.0833333, 0, 0.0625, 1]
  };

  void converter(double value, String from, String to) {
    int nFrom = measuresMap[from]!;
    int nTo = measuresMap[to]!;
    var multiplier = formulas[nFrom.toString()][nTo];

    if (multiplier == 0) {
```

```
    setState(() {
      errorMessage = 'Cannot perform this conversion';
    });
  } else {
    setState(() {
      errorMessage = '$value $from equals ${(value * multiplier).toStringAsFixed(2)} $to';
    });
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    body: SafeArea(
      child: Stack(
        children: [
          // Top decoration
          Positioned(
            top: 0,
            left: 0,
            right: 0,
            child: Container(
              height: 150,
              decoration: BoxDecoration(
                gradient: LinearGradient(
                  colors: [Colors.deepPurpleAccent, Colors.purpleAccent],
                  begin: Alignment.topCenter,
                  end: Alignment.bottomCenter,
                ),
                borderRadius: BorderRadius.only(
                  bottomLeft: Radius.circular(100),
                  bottomRight: Radius.circular(100),
                ),
              ),
            ),
          ),

          // Bottom decoration
          Positioned(
            bottom: 0,
            left: 0,
            right: 0,
            child: Container(
              height: 150,
    decoration: BoxDecoration(
      gradient: LinearGradient(
        colors: [Colors.purpleAccent, Colors.deepPurpleAccent],
        begin: Alignment.bottomCenter,
        end: Alignment.topCenter,
      ),
      borderRadius: BorderRadius.only(
        topLeft: Radius.circular(100),
        topRight: Radius.circular(100),
      ),
    ),
  ),
),

// Main content
Center(
  child: SingleChildScrollView(
    padding: const EdgeInsets.all(30),
    child: Container(
      padding: const EdgeInsets.all(30),
      decoration: BoxDecoration(
        color: Colors.white,
        borderRadius: BorderRadius.circular(20),
        boxShadow: [
          BoxShadow(
            color: Colors.grey.withOpacity(0.3),
            spreadRadius: 5,
            blurRadius: 15,
          ),
        ],
      ),
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          Text('Measure Converter', style: Theme.of(context).textTheme.headlineLarge),
          SizedBox(height: 30),
          TextField(
            style: TextStyle(fontSize: 20, color: Colors.black),
            decoration: InputDecoration(
              hintText: 'Enter Your Value',
              fillColor: Colors.grey[200],
            ),
            keyboardType: TextInputType.number,
            onChanged: (text) {
```

```
    var input = double.tryParse(text);
    if (input != null) {
      setState(() {
        _userInput = input;
      });
    }
  },
),
SizedBox(height: 30),
Row(
  mainAxisAlignment: MainAxisAlignment.spaceBetween,
  children: [
    _buildDropdown('From', _startValue, (String? value) {
      setState(() {
        _startValue = value;
      });
    }),
    _buildDropdown('To', _convertedMeasure, (String? value) {
      setState(() {
        _convertedMeasure = value;
      });
    }),
  ],
),
SizedBox(height: 30),
ElevatedButton(
  onPressed: () {
    if (_startValue != null &&
        _convertedMeasure != null &&
        _userInput != 0) {
      converter(_userInput, _startValue!, _convertedMeasure!);
    }
  },
  child: Text('Convert', style: Theme.of(context).textTheme.labelLarge),
),
SizedBox(height: 30),
if (errorMessage.isNotEmpty)
  Container(
    padding: EdgeInsets.all(20),
    decoration: BoxDecoration(
      color: Colors.grey[200],
      borderRadius: BorderRadius.circular(15),
    ),
    child: Text(
                        errorMessage,
                        style: TextStyle(fontSize: 24, color: Colors.deepPurpleAccent),
                        textAlign: TextAlign.center,
                      ),
                    ),
                  ],
                ),
              ),
            ),
          ),
        ],
      ),
    ),
  );
}

Widget _buildDropdown(String label, String? currentValue, ValueChanged<String?> onChanged) {
  return Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Text(label, style: TextStyle(fontSize: 20, color: Colors.deepPurpleAccent)),
      DropdownButton<String>(
        hint: Text('Choose Unit'),
        value: currentValue,
        items: fromUnits.map((String value) {
          return DropdownMenuItem<String>(
            value: value,
            child: Text(value),
          );
        }).toList(),
        onChanged: onChanged,
        dropdownColor: Colors.white,
        style: TextStyle(color: Colors.black, fontSize: 20),
      ),
    ],
  );
}
}
```