Project Report

On

**UNIVERSAL CONVERTOR**

Submitted in partial fulfilment of the requirements for the award of

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE & ENGINEERING**

(Artificial Intelligence & Machine Learning)

by

**Ms. I. Sivani**

**Ms. N. Divya Saahithya**

**Ms. D. Sai Lasya**

**Ms. K. Sreeja**

**Under the esteemed guidance of**

**Ms. S Annapoorna**

**Assistant Professor, CSE(AI&ML)**



**BVRIT HYDERABAD College of Engineering for Women**

**(AUTONOMOUS)**

**(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)**

**(NAAC Accredited - A Grade | NBA Accredited B.Tech. (EEE, ECE, CSE and IT)**

**Bachupally, Hyderabad – 500090**

2024-25

**BVRIT HYDERABAD**

**COLLEGE OF ENGINEERING FOR WOMEN**

**(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)**

**Accredited by NBA and NAAC with A Grade**

**Bachupally , Hyderabad – 500090**

**Department of Computer Science and Engineering**

**(Artificial Intelligence and Machine Learning)**



# CERTIFICATE

This is to certify that the project entitled **“Universal Converter”** is a bonafide work carried out by **Ms. I. Sivani (22WH1A6613), Ms. N. Divya Saahithya (22WH1A6616)**, **Ms. D. Sai Lasya (22WH1A6617), Ms. K. Sreeja (22WH1A6657**) in partial fulfillment for the award of B. Tech degree in **Computer Science & Engineering (AI&ML)**, **BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision. The results embodied in the project work have not been submitted to any other.

|  |  |
| --- | --- |
| **Internal Guide** | **Head of the Department** |
| **Ms. S. Annapoorna** | **Dr. B Lakshmi Praveena** |
| **Assistant Professor** | **HOD & Professor** |
| **Dept of CSE(AI&ML)** | **Dept of CSE(AI&ML)** |

# DECLARATION

We hereby declare that the work presented in this project entitled **“Universal Convertor ”** submitted towards completion of real time research project work in II Year of B.Tech of CSE(AI&ML) at **BVRIT HYDERABAD College of Engineering for Women,** Hyderabad is an authentic record of our original work carried out under the guidance of  **Ms. S.Annapoorna, Assistant Professor, Department of CSE(AI&ML).**

Sign with Date:

Ms. I Sivani

(22WH1A6613)

Sign with Date:

Ms. N Divya Saahithya

(22WH1A6616)

Sign with Date:

Ms. D Sai Lasya

(22WH1A6617)

Sign with Date:

Ms. K.Sreeja

(22WH1A6657)

**ACKNOWLEDGEMENT**

We would like to express our sincere thanks to **Dr. K V N Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women**, for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. B Lakshmi Praveena, Head of the Department, Department of CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our Internal Guide, **Ms.S.Annapoorna, Assistant Professor, CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for her constant guidance and encouragement throughout the project.

Finally, we would like to thank our project Coordinator, all Faculty and Staff of CSE(AI&ML) department who helped us directly or indirectly. Last but not least, we wish to acknowledge our **Parents and Friends** for giving moral strength and constant encouragement.

**Ms. I Sivani**

**(22WH1A6613)**

**Ms. N Divya Saahithya**

**(22WH1A6616)**

**Ms. D Sai Lasya**

**(22WH1A6617)**

**Ms. K Sreeja**

**(22WH1A6657)**

**ABSTRACT**

The Converter App is a cutting-edge Flutter-based mobile application meticulously designed to address the diverse conversion needs of modern users. With the growing demand for swift and accurate conversions in various fields, this app emerges as an all-in-one solution that combines efficiency, accuracy, and simplicity. The application encompasses three core functionalities: metric conversion, temperature conversion, and currency conversion.

The metric conversion module enables users to perform seamless transformations between various units, including length, weight, and volume, making it ideal for educational purposes, professional tasks, and everyday use. The temperature conversion feature simplifies the process of converting values across Celsius, Fahrenheit, and Kelvin scales, catering to both casual users and those in technical domains. The currency conversion module is equipped with real-time exchange rate updates, allowing users to effortlessly convert between global currencies, ensuring accuracy in financial calculations and aiding travellers, businesses, and students dealing with foreign currencies. By combining robust functionality with modern design principles, the Converter App stands out as an essential tool for individuals across different domains, fulfilling conversion requirements with speed and reliability.

**PROBLEM STATEMENT**

In today's globalized world, individuals frequently encounter the need to convert units, temperatures, and currencies. Existing tools are often fragmented, requiring users to switch between multiple apps or websites for specific conversions. This lack of an integrated solution leads to inefficiency and potential inaccuracies. The Converter App addresses this problem by offering a unified platform that supports metric, temperature, and currency conversions, ensuring convenience, accuracy, and real-time updates in a single application.

**FILE STRUCTURE**

home\_conv/

├── android/

├── build/

├── ios/

├── lib/

│ ├── curr\_conv/

│ │ └── currency.dart

│ ├── metric\_conv/

│ │ └── metricconv.dart

│ ├── temp\_conv/

│ │ └── temperature.dart

│ └── main.dart

├── linux/

├── macos/

├── test/

├── web/

├── windows/

├── .dart\_tool/

├── .idea/

├── .vscode/

├── .gitignore

├── .metadata

├── analysis\_options.yaml

├── home\_conv.iml

├── pubspec.lock

├── pubspec.yaml

└── README.md

**SOURCE CODE**

**Home + Buttons page :**

import 'package:flutter/material.dart';

void main() {

runApp(ConverterApp());

}

class ConverterApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Universal Converter',

theme: ThemeData(primarySwatch: Colors.blue),

home: HomePage(),

);

}

}

class HomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: Colors.blueGrey.shade50,

body: Center(

child: Padding(

padding: const EdgeInsets.symmetric(horizontal: 24.0, vertical: 40.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

crossAxisAlignment: CrossAxisAlignment.stretch,

children: [

Icon(

Icons.autorenew,

color: Colors.blue.shade400,

size: 100,

),

SizedBox(height: 20),

Text(

'Universal Converter',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 36,

fontWeight: FontWeight.bold,

color: Colors.blue.shade900,

),

),

SizedBox(height: 10),

Text(

'Convert currencies, metrics, temperature, and more!',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 18,

color: Colors.blue.shade700,

),

),

SizedBox(height: 40),

Divider(

color: Colors.blue.shade200,

thickness: 2,

indent: 50,

endIndent: 50,

),

SizedBox(height: 40),

SizedBox(

width: double.infinity,

child: ElevatedButton(

onPressed: () {

Navigator.push(

context,

MaterialPageRoute(

builder: (context) => ConverterSelectionPage()),

);

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.blue.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

shadowColor: Colors.blue.shade400,

elevation: 5,

),

child: Text(

'Get Started',

style: TextStyle(

fontSize: 20,

fontWeight: FontWeight.bold,

color: Colors.white,

),

),

),

),

],

),

),

),

);

}

}

// Converter Selection Page with centered and styled buttons

class ConverterSelectionPage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Choose Converter'),

backgroundColor: Colors.blue.shade700,

),

backgroundColor: Colors.blueGrey.shade50,

body: Center(

child: Padding(

padding: const EdgeInsets.all(24.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Text(

'Select a Conversion Type',

style: TextStyle(

fontSize: 24,

fontWeight: FontWeight.bold,

color: Colors.blue.shade900,

),

),

SizedBox(height: 30),

// Temperature Conversion Button

ElevatedButton.icon(

onPressed: () {

// Navigate to Temperature Converter Page

},

icon: Icon(Icons.thermostat, color: Colors.white),

label: Text(

'Temperature Converter',

style: TextStyle(fontSize: 18, color: Colors.white),

),

style: ElevatedButton.styleFrom(

minimumSize: Size(double.infinity, 50),

backgroundColor: Colors.indigo.shade600,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(20),

),

padding: EdgeInsets.symmetric(vertical: 15.0),

shadowColor: Colors.indigo.shade400,

elevation: 4,

),

),

SizedBox(height: 20),

// Metric Conversion Button

ElevatedButton.icon(

onPressed: () {

// Navigate to Metric Converter Page

},

icon: Icon(Icons.straighten, color: Colors.white),

label: Text(

'Metric Converter',

style: TextStyle(fontSize: 18, color: Colors.white),

),

style: ElevatedButton.styleFrom(

minimumSize: Size(double.infinity, 50),

backgroundColor: Colors.teal.shade600,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(20),

),

padding: EdgeInsets.symmetric(vertical: 15.0),

shadowColor: Colors.teal.shade400,

elevation: 4,

),

),

SizedBox(height: 20),

// Currency Conversion Button

ElevatedButton.icon(

onPressed: () {

// Navigate to Currency Converter Page

},

icon: Icon(Icons.attach\_money, color: Colors.white),

label: Text(

'Currency Converter',

style: TextStyle(fontSize: 18, color: Colors.white),

),

style: ElevatedButton.styleFrom(

minimumSize: Size(double.infinity, 50),

backgroundColor: Colors.green.shade600,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(20),

),

padding: EdgeInsets.symmetric(vertical: 15.0),

shadowColor: Colors.green.shade400,

elevation: 4,

),

),

],

),

),

),

);

}

}

**Metric Convertor:**

import 'package:flutter/material.dart';

void main() {

runApp(MetricConverterApp());

}

class MetricConverterApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Metric Converter',

theme: ThemeData(primarySwatch: Colors.deepPurple),

home: HomePage(),

);

}

}

class HomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: Colors.deepPurple.shade50,

body: Center(

child: Padding(

padding: const EdgeInsets.symmetric(horizontal: 24.0, vertical: 40.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

crossAxisAlignment: CrossAxisAlignment.stretch,

children: [

Icon(

Icons.straighten,

color: Colors.deepPurple.shade400,

size: 100,

),

SizedBox(height: 20),

Text(

'Metric Converter',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 32,

fontWeight: FontWeight.bold,

color: Colors.deepPurple.shade900,

),

),

SizedBox(height: 10),

Text(

'Convert units of length, area, volume, mass, and more.',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 18,

color: Colors.deepPurple.shade800,

),

),

SizedBox(height: 40),

ElevatedButton(

onPressed: () {

Navigator.push(

context,

MaterialPageRoute(builder: (context) => MetricConverterPage()),

);

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.deepPurple.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

child: Text(

'Start Converting',

style: TextStyle(

fontSize: 20,

fontWeight: FontWeight.bold,

color: Colors.white,

),

),

),

],

),

),

),

);

}

}

class MetricConverterPage extends StatefulWidget {

@override

\_MetricConverterPageState createState() => \_MetricConverterPageState();

}

class \_MetricConverterPageState extends State<MetricConverterPage> {

final TextEditingController \_amountController = TextEditingController();

String \_fromUnit = 'Kilometers';

String \_toUnit = 'Meters';

String \_result = '';

// Sample conversion rates for demonstration (1 unit in 'from' equals x in 'to')

final Map<String, Map<String, double>> \_conversionRates = {

'Length': {'Kilometers': 1000, 'Meters': 1, 'Centimeters': 0.01, 'Millimeters': 0.001},

'Area': {'Square Kilometers': 1e6, 'Square Meters': 1, 'Square Centimeters': 0.0001, 'Square Millimeters': 0.000001},

'Volume': {'Cubic Meters': 1, 'Liters': 0.001, 'Milliliters': 0.000001},

'Mass': {'Kilograms': 1, 'Grams': 0.001, 'Milligrams': 0.000001, 'Tons': 1000},

'Speed': {'Kilometers/Hour': 1, 'Meters/Second': 0.277778, 'Miles/Hour': 1.60934},

'Time': {'Hours': 3600, 'Minutes': 60, 'Seconds': 1},

'Weight': {'Kilograms': 1, 'Pounds': 0.453592, 'Ounces': 0.0283495},

};

String \_selectedCategory = 'Length';

void \_convertMetric() {

double amount = double.tryParse(\_amountController.text) ?? 0.0;

double fromRate = \_conversionRates[\_selectedCategory]![\_fromUnit]!;

double toRate = \_conversionRates[\_selectedCategory]![\_toUnit]!;

double convertedAmount = amount \* (fromRate / toRate);

setState(() {

\_result = '$amount $\_fromUnit = ${convertedAmount.toStringAsFixed(2)} $\_toUnit';

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Metric Converter'),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

// Category dropdown (Length, Area, etc.)

DropdownButton<String>(

value: \_selectedCategory,

items: \_conversionRates.keys.map((String category) {

return DropdownMenuItem<String>(

value: category,

child: Text(category),

);

}).toList(),

onChanged: (newCategory) {

setState(() {

\_selectedCategory = newCategory!;

\_fromUnit = \_conversionRates[\_selectedCategory]!.keys.first;

\_toUnit = \_conversionRates[\_selectedCategory]!.keys.last;

});

},

),

SizedBox(height: 16),

TextField(

controller: \_amountController,

decoration: InputDecoration(

labelText: 'Amount',

border: OutlineInputBorder(),

),

keyboardType: TextInputType.number,

),

SizedBox(height: 16),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

DropdownButton<String>(

value: \_fromUnit,

items: \_conversionRates[\_selectedCategory]!.keys.map((String unit) {

return DropdownMenuItem<String>(

value: unit,

child: Text(unit),

);

}).toList(),

onChanged: (newUnit) {

setState(() {

\_fromUnit = newUnit!;

});

},

),

Icon(Icons.arrow\_forward),

DropdownButton<String>(

value: \_toUnit,

items: \_conversionRates[\_selectedCategory]!.keys.map((String unit) {

return DropdownMenuItem<String>(

value: unit,

child: Text(unit),

);

}).toList(),

onChanged: (newUnit) {

setState(() {

\_toUnit = newUnit!;

});

},

),

],

),

SizedBox(height: 16),

ElevatedButton(

onPressed: \_convertMetric,

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.deepPurple.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

child: Text(

'Convert',

style: TextStyle(fontSize: 18, color: Colors.white),

),

),

SizedBox(height: 16),

Text(

\_result,

style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

),

],

),

),

);

}

}

**Currency Convertor:**

import 'package:flutter/material.dart';

void main() {

runApp(CurrencyConverterApp());

}

class CurrencyConverterApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Currency Converter',

theme: ThemeData(primarySwatch: Colors.green),

home: HomePage(),

);

}

}

class HomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: Colors.green.shade50, // Light background color

body: Center(

child: Padding(

padding: const EdgeInsets.symmetric(horizontal: 24.0, vertical: 40.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

crossAxisAlignment: CrossAxisAlignment.stretch,

children: [

Icon(

Icons.currency\_exchange,

color: Colors.green.shade400,

size: 100,

),

SizedBox(height: 20),

Text(

'Currency Converter',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 32,

fontWeight: FontWeight.bold,

color: Colors.green.shade900,

),

),

SizedBox(height: 10),

Text(

'Easily convert between popular currencies.',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 18,

color: Colors.green.shade800,

),

),

SizedBox(height: 40),

ElevatedButton(

onPressed: () {

Navigator.push(

context,

MaterialPageRoute(builder: (context) => CurrencyConverterPage()),

);

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.green.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

child: Text(

'Start Converting',

style: TextStyle(

fontSize: 20,

fontWeight: FontWeight.bold,

color: Colors.white,

),

),

),

],

),

),

),

);

}

}

class CurrencyConverterPage extends StatefulWidget {

@override

\_CurrencyConverterPageState createState() => \_CurrencyConverterPageState();

}

class \_CurrencyConverterPageState extends State<CurrencyConverterPage> {

final TextEditingController \_amountController = TextEditingController();

String \_fromCurrency = 'USD';

String \_toCurrency = 'INR';

double \_conversionRate = 74.5; // Example fixed conversion rate

String \_result = '';

void \_convertCurrency() {

double amount = double.tryParse(\_amountController.text) ?? 0.0;

double convertedAmount = amount \* \_conversionRate;

setState(() {

\_result = '$amount $\_fromCurrency = ${convertedAmount.toStringAsFixed(2)} $\_toCurrency';

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Currency Converter'),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

TextField(

controller: \_amountController,

decoration: InputDecoration(

labelText: 'Amount',

border: OutlineInputBorder(),

),

keyboardType: TextInputType.number,

),

SizedBox(height: 16),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

DropdownButton<String>(

value: \_fromCurrency,

items: <String>['USD', 'INR', 'GBP', 'EUR', 'JPY']

.map((String value) {

return DropdownMenuItem<String>(

value: value,

child: Text(value),

);

}).toList(),

onChanged: (newValue) {

setState(() {

\_fromCurrency = newValue!;

});

},

),

Icon(Icons.arrow\_forward),

DropdownButton<String>(

value: \_toCurrency,

items: <String>['USD', 'INR', 'GBP', 'EUR', 'JPY']

.map((String value) {

return DropdownMenuItem<String>(

value: value,

child: Text(value),

);

}).toList(),

onChanged: (newValue) {

setState(() {

\_toCurrency = newValue!;

});

},

),

],

),

SizedBox(height: 16),

ElevatedButton(

onPressed: \_convertCurrency,

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.green.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

child: Text(

'Convert',

style: TextStyle(fontSize: 18,color:Colors.white),

),

),

SizedBox(height: 16),

Text(

\_result,

style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

),

],

),

),

);

}

}

**Temperature Convertor:**

import 'package:flutter/material.dart';

void main() {

runApp(TemperatureConverterApp());

}

class TemperatureConverterApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Temperature Converter',

theme: ThemeData(primarySwatch: Colors.blue),

home: HomePage(),

);

}

}

class HomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: Colors.lightBlue.shade50,

body: Center(

child: Padding(

padding: const EdgeInsets.symmetric(horizontal: 24.0, vertical: 40.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

crossAxisAlignment: CrossAxisAlignment.stretch,

children: [

Icon(

Icons.thermostat,

color: Colors.blue.shade400,

size: 100,

),

SizedBox(height: 20),

Text(

'Temperature Converter',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 32,

fontWeight: FontWeight.bold,

color: Colors.blue.shade900,

),

),

SizedBox(height: 10),

Text(

'Convert temperatures between Celsius, Fahrenheit, Kelvin, Rankine, and Réaumur.',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 18,

color: Colors.blueGrey,

),

),

SizedBox(height: 40),

ElevatedButton(

onPressed: () {

Navigator.push(

context,

MaterialPageRoute(

builder: (context) => TemperatureConverterPage()),

);

},

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.blue.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

child: Text(

'Start Converting',

style: TextStyle(

fontSize: 20,

fontWeight: FontWeight.bold,

color: Colors.white,

),

),

),

],

),

),

),

);

}

}

class TemperatureConverterPage extends StatefulWidget {

@override

\_TemperatureConverterPageState createState() =>

\_TemperatureConverterPageState();

}

class \_TemperatureConverterPageState extends State<TemperatureConverterPage> {

final TextEditingController \_tempController = TextEditingController();

String \_fromUnit = 'Celsius';

String \_toUnit = 'Fahrenheit';

String \_result = '';

void \_convertTemperature() {

double temp = double.tryParse(\_tempController.text) ?? 0.0;

double convertedTemp;

if (\_fromUnit == \_toUnit) {

convertedTemp = temp;

} else if (\_fromUnit == 'Celsius' && \_toUnit == 'Fahrenheit') {

convertedTemp = (temp \* 9 / 5) + 32;

} else if (\_fromUnit == 'Celsius' && \_toUnit == 'Kelvin') {

convertedTemp = temp + 273.15;

} else if (\_fromUnit == 'Celsius' && \_toUnit == 'Rankine') {

convertedTemp = (temp + 273.15) \* 9 / 5;

} else if (\_fromUnit == 'Celsius' && \_toUnit == 'Réaumur') {

convertedTemp = temp \* 4 / 5;

} else if (\_fromUnit == 'Fahrenheit' && \_toUnit == 'Celsius') {

convertedTemp = (temp - 32) \* 5 / 9;

} else if (\_fromUnit == 'Fahrenheit' && \_toUnit == 'Kelvin') {

convertedTemp = (temp - 32) \* 5 / 9 + 273.15;

} else if (\_fromUnit == 'Fahrenheit' && \_toUnit == 'Rankine') {

convertedTemp = temp + 459.67;

} else if (\_fromUnit == 'Fahrenheit' && \_toUnit == 'Réaumur') {

convertedTemp = (temp - 32) \* 4 / 9;

} else if (\_fromUnit == 'Kelvin' && \_toUnit == 'Celsius') {

convertedTemp = temp - 273.15;

} else if (\_fromUnit == 'Kelvin' && \_toUnit == 'Fahrenheit') {

convertedTemp = (temp - 273.15) \* 9 / 5 + 32;

} else if (\_fromUnit == 'Kelvin' && \_toUnit == 'Rankine') {

convertedTemp = temp \* 9 / 5;

} else if (\_fromUnit == 'Kelvin' && \_toUnit == 'Réaumur') {

convertedTemp = (temp - 273.15) \* 4 / 5;

} else if (\_fromUnit == 'Rankine' && \_toUnit == 'Celsius') {

convertedTemp = (temp - 491.67) \* 5 / 9;

} else if (\_fromUnit == 'Rankine' && \_toUnit == 'Fahrenheit') {

convertedTemp = temp - 459.67;

} else if (\_fromUnit == 'Rankine' && \_toUnit == 'Kelvin') {

convertedTemp = temp \* 5 / 9;

} else if (\_fromUnit == 'Rankine' && \_toUnit == 'Réaumur') {

convertedTemp = (temp - 491.67) \* 4 / 9;

} else if (\_fromUnit == 'Réaumur' && \_toUnit == 'Celsius') {

convertedTemp = temp \* 5 / 4;

} else if (\_fromUnit == 'Réaumur' && \_toUnit == 'Fahrenheit') {

convertedTemp = (temp \* 9 / 4) + 32;

} else if (\_fromUnit == 'Réaumur' && \_toUnit == 'Kelvin') {

convertedTemp = temp \* 5 / 4 + 273.15;

} else if (\_fromUnit == 'Réaumur' && \_toUnit == 'Rankine') {

convertedTemp = (temp \* 9 / 4) + 491.67;

} else {

convertedTemp = temp; // Same unit conversion

}

setState(() {

\_result = '$temp $\_fromUnit = ${convertedTemp.toStringAsFixed(2)} $\_toUnit';

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Temperature Converter'),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

TextField(

controller: \_tempController,

decoration: InputDecoration(

labelText: 'Temperature',

border: OutlineInputBorder(),

),

keyboardType: TextInputType.number,

),

SizedBox(height: 16),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

DropdownButton<String>(

value: \_fromUnit,

items: <String>['Celsius', 'Fahrenheit', 'Kelvin', 'Rankine', 'Réaumur']

.map((String value) {

return DropdownMenuItem<String>(

value: value,

child: Text(value),

);

}).toList(),

onChanged: (newValue) {

setState(() {

\_fromUnit = newValue!;

});

},

),

Icon(Icons.arrow\_forward),

DropdownButton<String>(

value: \_toUnit,

items: <String>['Celsius', 'Fahrenheit', 'Kelvin', 'Rankine', 'Réaumur']

.map((String value) {

return DropdownMenuItem<String>(

value: value,

child: Text(value),

);

}).toList(),

onChanged: (newValue) {

setState(() {

\_toUnit = newValue!;

});

},

),

],

),

SizedBox(height: 16),

ElevatedButton(

onPressed: \_convertTemperature,

style: ElevatedButton.styleFrom(

padding: EdgeInsets.symmetric(vertical: 16.0),

backgroundColor: Colors.blue.shade700,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

child: Text(

'Convert',

style: TextStyle(fontSize: 18,color:Colors.white),

),

),

SizedBox(height: 16),

Text(

\_result,

style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

),

],

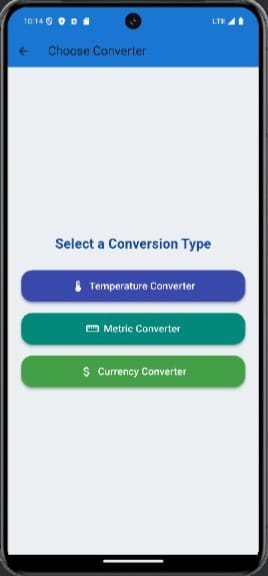
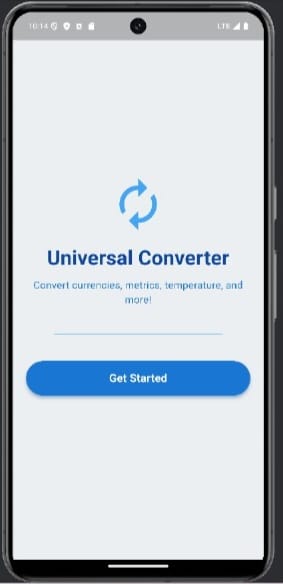
),

),

);

}}

**OUTPUT**

****