K S RANGASAMY COLLEGE OF TECHNOLOGY

(Autonomous)

TIRUCHENGODE-637215

A MINI PROJECT REPORT

UNIT CONVERTER

60 IT L04 – C# AND .NET FRAMEWORK

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

Submitted by

SIVASANKARI A (73772214208)

K S RANGASAMY COLLEGE OF TECHNOLOGY

(Autonomous)

TIRUCHENGODE-637215

BONAFIDE CERTIFICATE

Certified that this project report titled "**UNIT CONVERTER**" is the bonafide work of **S I V A S A N K A R I A (73772214208)** who carried out the project under my guidance.

Dr. S. Madhavi M.E., Ph.D.,
Professor and Head of the Department
Department of Computer Science and Engineering
K.S. Rangasamy College of Technology
Tiruchengode-637 215.

Mrs. N. Sathiyapriya M.E.,
Assistant Professor
Department of Information Technology
K.S. Rangasamy College of Technology
Tiruchengode - 637 215.

ABSTRACT

Unit Converter is a simple yet versatile application designed to facilitate the conversion of different units of measurement, such as length and temperature, between standard units. The program provides an interactive menu-driven interface that allows users to select the type of conversion they wish to perform, including length conversions (e.g., meters to kilometers, inches to feet) and temperature conversions (e.g., Celsius to Fahrenheit, Fahrenheit to Celsius). The tool is aimed at providing an easy and quick solution for individuals or professionals needing to perform unit conversions frequently.

This application is implemented in C# as a console-based program, ensuring a straightforward and user-friendly experience. With the flexibility to add more conversion types, such as weight, volume, or currency, the Unit Converter can be expanded for broader use cases. The system promotes accuracy and simplicity, making it a useful tool for students, engineers, scientists, or anyone requiring quick unit conversions for everyday tasks.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Unit conversion is a fundamental task in various fields, such as science, engineering, and everyday life, where measurements need to be converted from one unit to another. With the increasing complexity of international standards and diverse unit systems, having a reliable tool for these conversions becomes essential. The **Unit Converter** application aims to address this need by providing a simple and intuitive solution for converting between different units of measurement, such as length, temperature, weight, and more.

Developed in **C#**, the Unit Converter is a console-based application that allows users to easily convert values between various units. The program provides an interactive, menu-driven interface where users can select the type of conversion they require and input the corresponding value. The application is designed to be extensible, allowing additional conversion functionalities to be added as needed. It simplifies the process of performing unit conversions, making it a practical tool for professionals, students, or anyone who frequently needs to work with different units of measurement.

CHAPTER 2

REQUIREMENT ANALYSIS

Functional Requirements:

1. **Add a Conversion**: Users can input values for different units (e.g., length, temperature) and select the conversion type (e.g., meters to kilometers, Celsius to Fahrenheit).
2. **Edit a Conversion**: Users can modify the conversion values or change the unit types for existing conversions.
3. **Clear Conversion Fields**: Input fields can be reset, allowing users to enter new values or start fresh with a different unit conversion.
4. **Save Conversion History**: The application keeps track of previous conversions, allowing users to view or repeat past calculations.
5. **Display Results in Real-Time**: The conversion results are shown instantly after users input the values, ensuring smooth and immediate feedback.

Non-Functional Requirements:

1. **User-Friendly Interface**: The application provides a simple, clean, and easy-to-navigate interface, ensuring a smooth user experience for anyone performing unit conversions.
2. **Responsiveness**: All operations, such as unit conversions and result display, execute quickly, ensuring that users get real-time feedback without delays.

3. **Scalability**: The application is built to accommodate future expansion, such as adding more unit types, currency conversion, or advanced features like saving preferences or historical data.

Tools and Technologies:

- **Programming Language**: C#
- **Framework**: .NET Framework (Windows Forms)
- **Integrated Development Environment (IDE)**: Visual Studio

CHAPTER 3

DESIGN AND IMPLEMENTATION

System Overview :

Graphical User Interface (GUI):

1. **Data Display**: The conversion results are displayed in a clear, readable format, with the input values and results shown in separate fields for easy comparison.
2. **Input Fields (TextBox)**: TextBoxes are used for users to input the values they want to convert, along with selecting the units for conversion (e.g., length, temperature).
3. **Buttons (Button)**: Buttons trigger actions such as converting values, clearing the fields, and resetting the application.
4. **ComboBoxes**: Dropdown menus are used for users to select the units for conversion (e.g., from "meters" to "kilometers", or from "Celsius" to "Fahrenheit").

Data Management:

1. **Conversion Logic**: A central method or class performs the unit conversions based on the selected unit types and input values.
2. **Real-Time Feedback**: As the user inputs values, the conversion result is displayed instantly in the GUI without requiring a manual refresh.

Code Summary:

- **Form Load**: Initializes the conversion logic, including any default unit selections or previously saved preferences.
- **Unit Conversion**: When a user selects the conversion type and inputs the value, the program calculates and displays the result in real time.
- **Clear Fields**: Clears the input fields for easy entry of new conversion values.
- **Save History**: Optionally saves previous conversions for later reference, allowing users to view or reuse previous results.
- **Handle Conversion Errors**: Displays error messages for invalid inputs (e.g., entering non-numeric values).

CHAPTER 4

CODE

```
using System;



namespace UnitConverter
{
    class Program
    {
        static void Main(string[] args)
        {
            while (true)
            {
                Console.Clear();
                Console.WriteLine("=== Unit Converter ===");
                Console.WriteLine("1. Length Conversion");
                Console.WriteLine("2. Temperature Conversion");
                Console.WriteLine("3. Exit");
                Console.Write("Choose an option (1-3): ");
```

```
int choice = int.Parse(Console.ReadLine());


switch (choice)
{
    case 1:
        LengthConversion();
        break;
    case 2:
        TemperatureConversion();
        break;
    case 3:
        Console.WriteLine("Exiting the program. Goodbye!");
        return;
    default:
        Console.WriteLine("Invalid choice. Please try again.");
        break;
}
```

```
        Console.WriteLine("\nPress any key to continue...");
        Console.ReadKey();
    }
}

static void LengthConversion()
{
    Console.Clear();
    Console.WriteLine("=== Length Conversion ===");
    Console.WriteLine("1. Meters to Kilometers");
    Console.WriteLine("2. Inches to Feet");
    Console.Write("Choose an option (1-2): ");
    int choice = int.Parse(Console.ReadLine());

    switch (choice)
    {
        case 1:
            Console.Write("Enter length in meters: ");
```

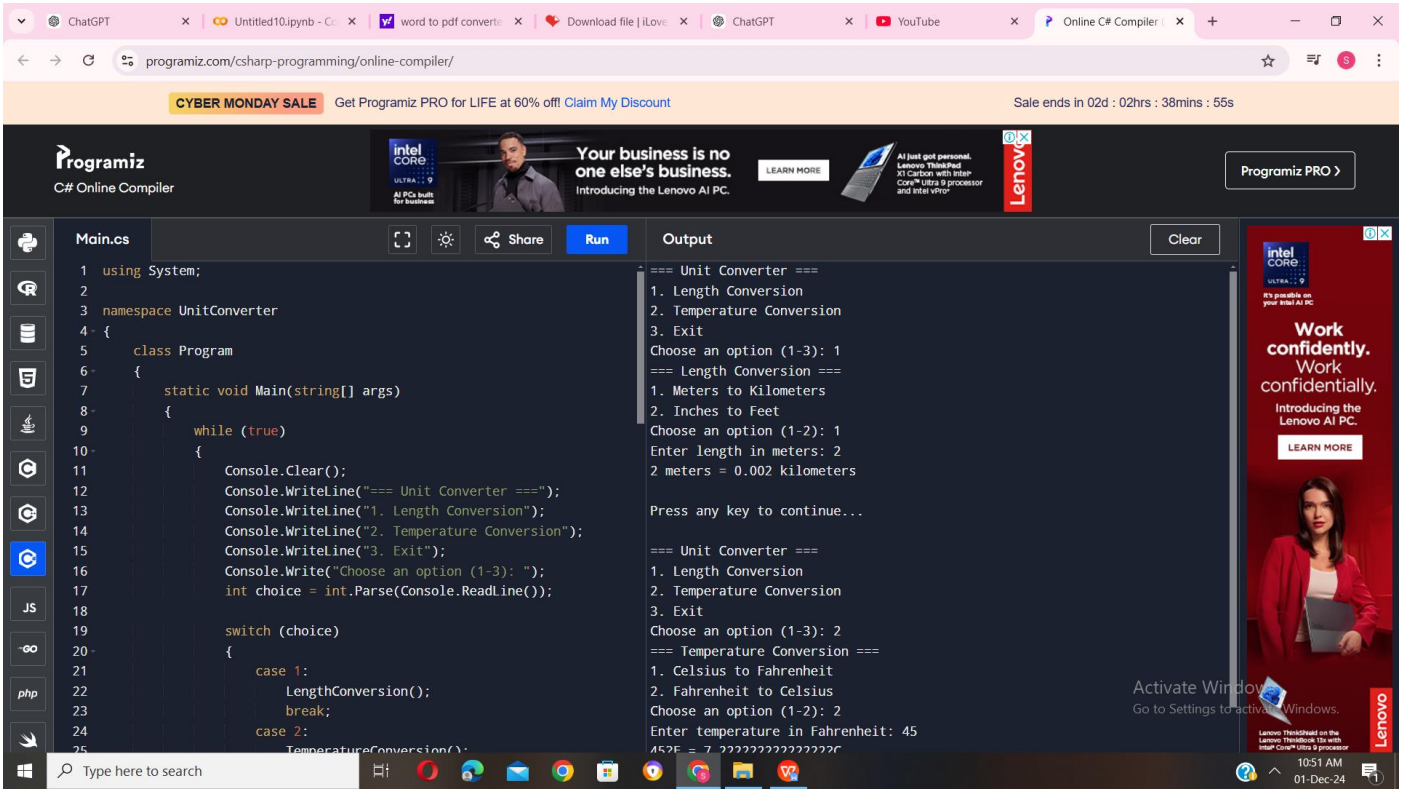
```
                double meters = double.Parse(Console.ReadLine());
                Console.WriteLine($"{meters} meters = {meters / 1000}
kilometers");
                break;
            case 2:
                Console.Write("Enter length in inches: ");
                double inches = double.Parse(Console.ReadLine());
                Console.WriteLine($"{inches} inches = {inches / 12} feet");
                break;
            default:
                Console.WriteLine("Invalid choice.");
                break;
        }
    }

    static void TemperatureConversion()
    {
        Console.Clear();
```

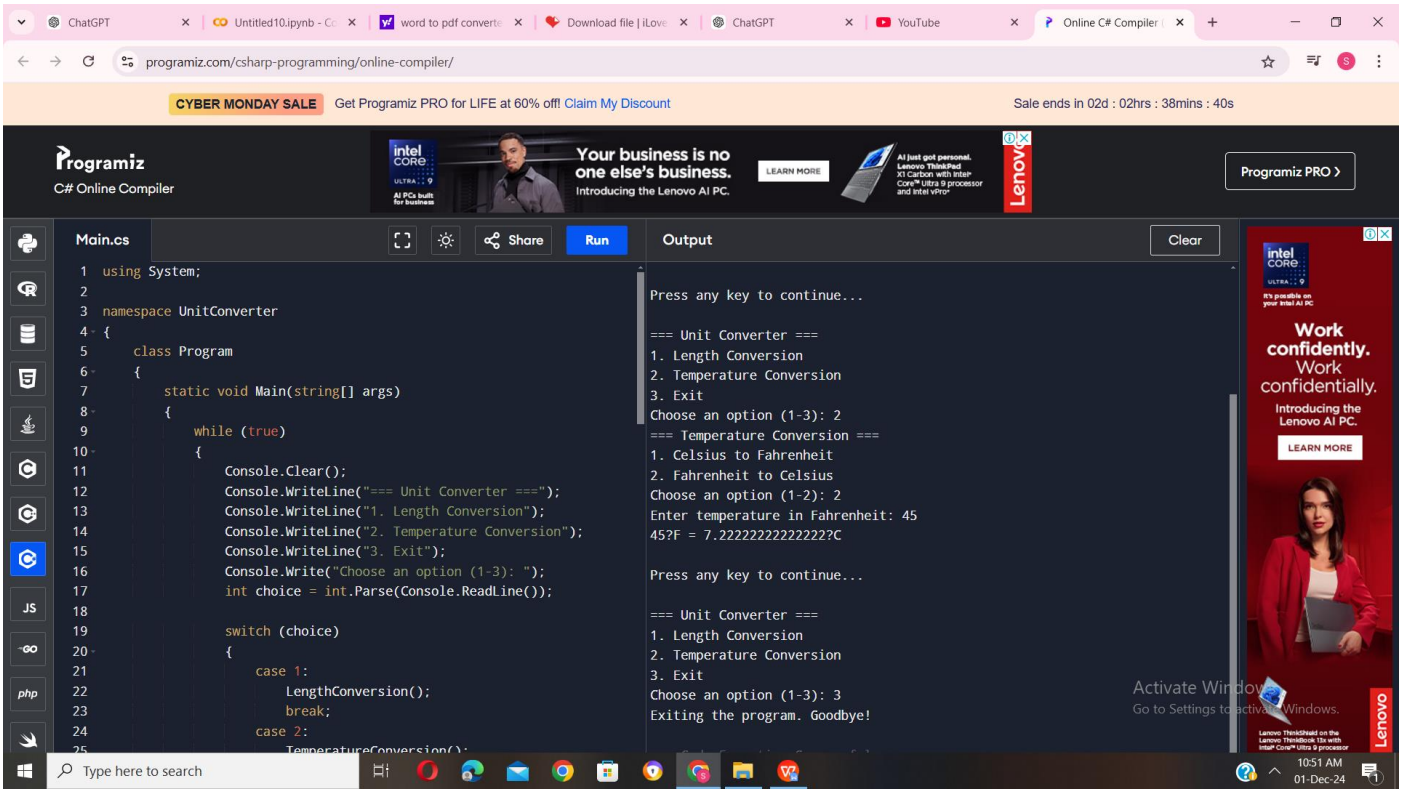
```
        Console.WriteLine("=== Temperature Conversion ===");
        Console.WriteLine("1. Celsius to Fahrenheit");
        Console.WriteLine("2. Fahrenheit to Celsius");
        Console.Write("Choose an option (1-2): ");
        int choice = int.Parse(Console.ReadLine());

        switch (choice)
        {
            case 1:
                Console.Write("Enter temperature in Celsius: ");
                double celsius = double.Parse(Console.ReadLine());
                Console.WriteLine($"{celsius}°C = {(celsius * 9 / 5) + 32}°F");
                break;
            case 2:
                Console.Write("Enter temperature in Fahrenheit: ");
                double fahrenheit = double.Parse(Console.ReadLine());
                Console.WriteLine($"{fahrenheit}°F = {(fahrenheit - 32) * 5 /
9}°C");
```

```
                break;

            default:

                Console.WriteLine("Invalid choice.");

                break;

        }

    }

}
```

CHAPTER 5

OUTPUT





CHAPTER 6

CONCLUSION

In conclusion, the **Unit Converter** application provides a simple, efficient, and user-friendly solution for converting various units of measurement. By leveraging a clear graphical user interface and real-time feedback, the application ensures a smooth user experience. The system's design allows for easy expansion, making it adaptable to different types of conversions and additional features like conversion history tracking. Overall, the Unit Converter serves as a valuable tool for anyone needing quick and accurate unit conversions, whether for educational, professional, or everyday use.

CHAPTER 7

REFERENCES

- Microsoft Documentation. (n.d.). *Windows Forms Controls*. Retrieved from https://docs.microsoft.com/en-us/dotnet/desktop/winforms/controls/
- Microsoft Documentation. (n.d.). *C# Programming Guide*. Retrieved from https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/
- C# Tutorials. (n.d.). *How to use DataGridView in C# Windows Forms*. Retrieved from https://www.c-sharpcorner.com/article/how-to-use-datagridview-in-c-sharp/