**Northern University of Business and Technology Khulna**

**Department of Computer Science and Engineering**

**[Project Report]**

Project Title:

Unit Converter (Temperature, Length, Weight, etc.)

Submitted by:

Ratul Raha

11220320949

ratulraha.04@gmail.com

Course Title: Software Development I

Course Code: CSE 2216

Section: 4E

Submitted to:

Shovon Mandal [SM]

Lecturer,

Northern University of Business and Technology Khulna

**Date of Submission: 21.08.2025**

Contents

[2. Abstract: 1](#_Toc204954373)

[3. Introduction: 1](#_Toc204954374)

[4. Literature Review: 1](#_Toc204954375)

[5. Methodology: 1](#_Toc204954376)

[6. Implementation: 1](#_Toc204954377)

[7. Results and Analysis: 2](#_Toc204954378)

[8. Discussion: 3](#_Toc204954379)

[9. Conclusion: 3](#_Toc204954380)

[10. References: 3](#_Toc204954381)

**Table of Figure:**

[Figure 1: Flowchart Template (Replace with your own flowchart name) 1](#_Toc204954367)

[Figure 2: Create board method (Replace with your own flowchart name) 2](#_Toc204954368)

[Figure 3: Output of Tic Tac Toe Step 1 (Replace with your own flowchart name) 2](#_Toc204954369)

[Figure 4:Output of Tic Tac Toe Step 2 (Replace with your own flowchart name) 2](#_Toc204954370)

Student Management System

# 2. Abstract:

This project is a web-based Unit Converter for temperature, length, and weight. Built with HTML, CSS, and Vanilla JavaScript, it provides quick and accurate conversions through a simple and responsive interface. The system ensures ease of use and reliable results for everyday measurement needs.

# 3. Introduction:

**Background:**  
Converting units like temperature, length, and weight is common in daily life and various fields. Manual conversions can be slow and error-prone.

**Problem Statement:**  
People often struggle to convert units quickly and accurately without a simple tool.

**Objectives:**

* Create an easy-to-use converter for temperature, length, and weight units.
* Provide real-time, accurate conversion results.

**Scope:**  
Supports common units in temperature, length, and weight via a web-based interface using JavaScript. Complex or uncommon units are not included.

# 4. Literature Review:

Unit converters have progressed from manual methods to digital tools that offer fast and accurate conversions. Popular apps like Engineering Unit Converter are widely used worldwide. Research shows many users, especially students, find unit conversions challenging, highlighting the need for simple and reliable tools. Developers create user-friendly applications using programming techniques to ensure accuracy and ease of use.

**References:**  
[1]<https://journals.flvc.org/cee/article/view/116477/122664>

[2]<https://files.eric.ed.gov/fulltext/EJ1190540.pdf>

[3]<https://ijarsct.co.in/Paper17613.pdf>

[4] <https://www.cs.utexas.edu/~novak/units95.html>

# 5. Methodology:

**Approach:**  
The project was developed as a web-based unit converter to provide instant and accurate conversions for temperature, length, and weight. The interface allows users to select the unit type, input a value, and view real-time results through JavaScript-based logic.

**Algorithms:**

* Conditional checks and mathematical formulas for temperature conversions (Celsius, Fahrenheit, Kelvin).
* Multiplicative conversion factors for length and weight conversions using reference unit values.
* Event-driven updates to trigger conversions instantly when input or selection changes.

**Tools and Technologies:**

* **Frontend:** HTML5, CSS3, JavaScript (Vanilla JS)
* **Development Environment:** Web Browser, Text Editor (VS Code)
* **Platform:** Any modern operating system with browser support.

**Experimental Design:**  
The converter was tested with known unit conversion values to verify accuracy. Multiple test cases were conducted for each unit category, including same-unit conversions (result equals input) and cross-unit conversions. Edge cases such as invalid or empty inputs were also checked to ensure proper error handling.

**Flowchart:**

A diagram of a flowchart

AI-generated content may be incorrect.

Figure 1: Flowchart *Unit Converter (Temperature, Length, Weight, etc*)

**Description:**

The flowchart begins with the user selecting the type of conversion—temperature, length, or weight. The user then inputs a value along with the source and target units. The program checks if both units are the same; if so, it returns the original value. Otherwise, it applies the appropriate conversion formula or rate based on the chosen type. The converted result is then displayed, and the user can choose to perform another conversion or end the process.

# 6. Implementation:

**System Architecture:**  
The Unit Converter follows a **client-side web application** architecture. The user interface (UI) is built with HTML and CSS, while JavaScript handles the logic, conversions, and DOM updates. No server or database is required, as all operations are performed locally in the browser.

**Components:**

1. **User Interface (UI):** HTML form elements for selecting conversion type, units, and entering values.
2. **Conversion Logic:** JavaScript functions that calculate conversions using formulas or predefined unit rates.
3. **Event Handling:** Event listeners that trigger conversions when user input changes.
4. **Output Display:** A dynamic result field that updates in real time.

**Code Excerpts:**  
Example of the **conversion logic** for temperature:

A computer screen with white text

AI-generated content may be incorrect.

Figure 2 : Unit Conversion Process Program Flowchart

**Description:**

This flowchart illustrates the process of converting units in the application. It starts with user input (type, from-unit, to-unit, and value), applies the appropriate conversion formulas or rates, and displays the result. If units are the same, the original value is returned without calculation. The process can repeat until the user exits.

# 7. Results and Analysis:

**Results:**  
The Unit Converter successfully performs accurate conversions for temperature, length, and weight units. The results are displayed in real time as users change inputs or unit selections. Testing with known values confirmed the correctness of conversion formulas and rates.

**Analysis:**  
The project meets its objectives by providing a simple, fast, and user-friendly conversion tool. Real-time conversion improves usability, and the JavaScript logic ensures precise results without requiring server-side processing. This solution is efficient for common daily conversion needs but could be extended to support additional units or categories in the future.

**Snippet:**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3: screenshot of my Unit Converter interface showing an example conversion

**Description:**This figure shows the application’s output screen, where the user has selected the conversion type, chosen units, entered a value, and received the converted result instantly.

# 8. Discussion:

**Interpretation:**  
The results show that the Unit Converter achieves its main goal—providing quick, accurate, and user-friendly conversions for temperature, length, and weight. The application effectively addresses the problem of manual conversion errors by automating the process in real time.

**Challenges:**

* Designing an interface that is both simple and responsive.
* Ensuring accuracy of formulas and conversion rates.
* Handling edge cases like empty input, invalid values, or same-unit conversions.

**Limitations:**

* Only supports three categories (temperature, length, weight).
* No offline mobile app version—works only in a web browser.
* Does not handle very large or scientific unit systems (e.g., astronomical units, nanometers).

# 9. Conclusion:

**Summary:**  
This project developed a web-based Unit Converter capable of converting between temperature, length, and weight units. Using HTML, CSS, and JavaScript, the tool provides accurate, real-time results through an interactive interface. The solution eliminates manual conversion errors and improves user convenience.

**Contributions:**

* Created a lightweight, browser-based application with no need for server-side processing.
* Implemented accurate conversion formulas and rates for three major categories.
* Designed a user-friendly interface with instant updates based on input changes.

**Future Work:**

* Add more unit categories such as volume, area, speed, and time.
* Implement a responsive mobile-friendly design.
* Include a history log for previous conversions.
* Create an offline mobile app version for Android and iOS.

# 10. References:

[1] W3Schools.com. *JavaScript HTML DOM*. Available at: <https://www.w3schools.com/js/js_htmldom.asp>

[2] W3Schools.com. *JavaScript Functions*. Available at: <https://www.w3schools.com/js/js_functions.asp>

[3] W3Schools.com. *JavaScript Events*. Available at: <https://www.w3schools.com/js/js_events.asp>

[4] Journals.flvc.org. *Engineering Unit Converter App Usage Statistics*. Available at: <https://journals.flvc.org/cee/article/view/116477/122664>

[5] ERIC. *Study on Metric Unit Conversion Challenges*. Available at: <https://files.eric.ed.gov/fulltext/EJ1190540.pdf>

[6] IJARSCT. *Android Unit Converter Development*. Available at: <https://ijarsct.co.in/Paper17613.pdf>

[7] Novak, G.S. *Software Unit Conversion Methods*. Available at: <https://www.cs.utexas.edu/~novak/units95.html>