#### Submitted By: M. Talha Atif F2021266573

#### Assignment 4

# Code:

<!DOCTYPE html>

<html lang="en">

    <!-- content -->

<head>

    <title>Unit Converter Created By Talha Atif</title>

    <style>

          /\* styling of html page\*/

body {

    /\* here flex with align-item and justify-content is used to align divs of body at center \*/

    font-family: Arial, sans-serif;

    background-color: #f4f4f4;

    display: flex;

    font-size: 20px;

    justify-content: center;

    align-items: center;

    height: 100vh;

    margin: 0;

}

.unitConverter {

    background: white;

    padding: 20px;

    border-radius: 8px;

    box-shadow: 0 0 10px rgba(206, 51, 51, 0.1);

    text-align: center;

}

h1 {

    margin-bottom: 20px;

}

input, select, button {

    margin: 9px 0;

    padding: 8px;

    border-radius: 5px;

    border: 1px solid #c02c2c;

    font-size: 10px;

}

button {

    background-color: #ff8800;

    color: white;

    border: none;

    cursor: pointer;

}

button:hover {

    background-color: #e7b249;

}

#result {

    margin-top: 20px;

    padding: 10px;

    background-color: #e9ecef;

    border-radius: 4px;

    color:#c02c2c;

    font-weight: bold;

    font-size: 70px;

}

    </style>

</head>

<body>

    <!-- div having all tags -->

    <div class="unitConverter">

        <!--  heading tag -->

        <h1>Unit Converter By Talha Atif</h1>

        <!--  input for number by user -->

        <input type="number" id="value" placeholder="Hey there! Enter value to convert">

        <!-- selction for source unit -->

        <select id="fromUnit">

            <!-- optgroup is used to make grouping of units here, Length for length grouping and Weight for weight -->

            <optgroup label="Length">

                <option value="m">Meters (m)</option>

                <option value="km">Kilometers (km)</option>

                <option value="mi">Miles (mi)</option>

                <option value="ft">Feet (ft)</option>

            </optgroup>

            <!-- optgroup is used to make grouping of units here, Length for length grouping and Weight for weight -->

            <optgroup label="Weight">

                <option value="g">Grams (g)</option>

                <option value="kg">Kilograms (kg)</option>

                <option value="oz">Ounces (oz)</option>

                <option value="lb">Pounds (lb)</option>

            </optgroup>

        </select>

        <!-- selection for target unit -->

        <select id="toUnit">

            <!-- optgroup is used to make grouping of units here, Length for length grouping and Weight for weight -->

            <optgroup label="Length">

                <option value="m">Meters (m)</option>

                <option value="km">Kilometers (km)</option>

                <option value="mi">Miles (mi)</option>

                <option value="ft">Feet (ft)</option>

            </optgroup>

            <!-- optgroup is used to make grouping of units here, Length for length grouping and Weight for weight -->

            <optgroup label="Weight">

                <option value="g">Grams (g)</option>

                <option value="kg">Kilograms (kg)</option>

                <option value="oz">Ounces (oz)</option>

                <option value="lb">Pounds (lb)</option>

            </optgroup>

        </select>

        <!-- When use will click this button, conversion will be done -->

        <button id="convertBtn" onclick="convertUnits()">Convert Value</button>

        <!-- Dive For storing result -->

        <div id="result"></div>

    </div>

    <!-- code for behaviour -->

    <script>

function convertUnits(){

    //  objects are used here which are actually dictionaries you can say or even map of keys

    // here units are as keys and values are convertion rates

const lengthToBase = {

    m: 1,

    km: 1000,

    mi: 1609.34,

    ft: 0.3048

};

// here units are as keys and values are convertion rates

const baseToLength = {

    m: 1,

    km: 0.001,

    mi: 0.000621371,

    ft: 3.28084

};

// here units are as keys and values are convertion rates

const weightToBase = {

    g: 1,

    kg: 1000,

    oz: 28.3495,

    lb: 453.592

};

// here units are as keys and values are convertion rates

const baseToWeight = {

    g: 1,

    kg: 0.001,

    oz: 0.035274,

    lb: 0.00220462

}

//  get number and cast into float for more precision

    const value = parseFloat(document.getElementById('value').value);

    // get source and target uni

    const fromUnit = document.getElementById('fromUnit').value;

    const toUnit = document.getElementById('toUnit').value;

    // isNan function will return false if value obtained from user input is a number else true for not a number

    if (isNaN(value) || value<0) {

        document.getElementById('result').textContent = 'Please enter a valid number.';

        return;

    }

    //  create variable with let for storing calculated result

    let covertedResultAfterUnits;

    // if source and target unit are from length class then do accordingly

    if (lengthToBase[fromUnit] && baseToLength[toUnit]) {

        // Convert length

        const valueInBase = value \* lengthToBase[fromUnit]; // Convert to base (meters)

        covertedResultAfterUnits = valueInBase \* baseToLength[toUnit]; // Convert from base to target unit

        document.getElementById('result').style.color = 'green';

    }

    // if source and target unit are from weight class then do accordingly

    else if (weightToBase[fromUnit] && baseToWeight[toUnit]) {

        // Convert weight

        const valueInBase = value \* weightToBase[fromUnit]; // Convert to base (grams)

        covertedResultAfterUnits = valueInBase \* baseToWeight[toUnit]; // Convert from base to target unit

        document.getElementById('result').style.color = 'blue';

    } else {

        document.getElementById('result').innerHTML = 'Invalid unit conversion.';

        document.getElementById('result').style.color = 'red';

        return;

    }

    document.getElementById('result').innerHTML= `${value} ${fromUnit} = ${covertedResultAfterUnits} ${toUnit}`;

}

    </script>

    <!--  end of script tag -->

</body> <!--  end of body tag -->

<!--  end of html tag -->

</html>

# Output:











  

