

Income tax Calculator

By Pranav Talwar (000368374)
& Avi Pancholi(000367122)

1. Introduction

This project is an implementation of a simple **Income Tax Calculator** that allows users to enter their taxable income and instantly calculates the tax they owe based on predefined income brackets. The project focuses on delivering an easy-to-use interface that simplifies tax computation. By using this calculator, users can quickly determine their tax obligations, promoting better financial planning and decision-making.

2. Objective

The primary objective of the Income Tax Calculator is to provide a user-friendly application that helps individuals compute their income tax with minimal effort. The calculator applies different tax rates to income brackets, ensuring that the tax owed is calculated accurately based on the user's taxable income.

3. Functional Requirements

The calculator performs the following key functions:

- Accepts user input for taxable income.
 - Applies different tax rates depending on the income bracket.
 - Displays the calculated tax owed to the user.
-

4. System Design

4.1 Front-End Design

The front end of the application consists of a simple, clean, and centered layout. The interface includes:

- An input field for the user to enter their taxable income.
- A button to trigger the tax calculation.
- A display area to show the calculated tax owed.

HTML Structure:

The application uses a structured layout with the following main components:

1. **Input Section:** A labeled input field for users to enter their taxable income.
2. **Calculate Button:** A button that triggers the tax calculation.
3. **Result Section:** A display area where the calculated tax owed is shown.

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Income Tax Calculator</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>
```

Linking CSS file with Html

```
<section class="calculator">

    <div class="input">

        <label >Enter taxable income:</label>

        <input placeholder="Income" id="income">

        <button id="btn">Calculate</button>

    </div>

    <div class="result">
        <label >Income Tax Owed:</label>

        <span id="show">0.00</span>

    </div>

</section>

<script src="script.js"></script>

</body>

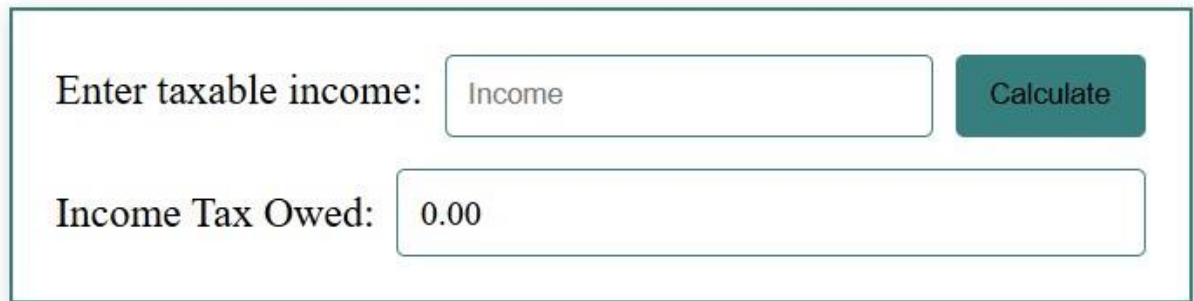
</html>
```

Linking JS file, and main body of the webpage

4.2 CSS Styling

The interface is styled using CSS to provide a clean and responsive design. Key elements include:

- A centered calculator container with a teal border and shadow effect to enhance the layout.
- Input fields, labels, and buttons are styled for better user interaction, including hover effects on the buttons.

A screenshot of a web-based income tax calculator. The interface is contained within a teal-bordered box with a subtle shadow. It features two rows of input fields. The first row has the label "Enter taxable income:" followed by a text input field containing the word "Income" and a teal "Calculate" button. The second row has the label "Income Tax Owed:" followed by a text input field containing the value "0.00".

Enter taxable income: Calculate

Income Tax Owed:

Figure 1 Program

The CSS file defines the layout of the input fields, button, and result section, ensuring the application is visually appealing and easy to use.

```
.calculator {  
  
    border: 2px solid teal;  
  
    padding: 20px;  
  
    width: 90%;  
  
    max-width: 500px;  
  
    margin: 50px auto;  
  
    text-align: left;  
  
  
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.155);  
  
}
```

```
.input {  
  
    margin-bottom: 15px;  
  
    display: flex;  
  
    justify-content: center;  
}  
  
.input input {  
  
    flex: 1;  
  
    padding: 10px;  
  
    border: 1px solid teal;  
  
    border-radius: 4px;  
  
    margin-right: 10px;  
}  
  
.input label {  
  
    margin-right: 10px;  
  
    margin-top: 6px;  
  
    font-size: 1.2em;  
}
```

```
.input button {

    padding: 10px 15px;

    border: 1px solid teal;

    border-radius: 4px;

}

.input button:hover {

    background-color: teal;

}

.result {

    margin-top: 15px;

    display: flex;

    align-items: center;

}

.result label {

    margin-right: 10px;

    font-size: 1.2em;

}

.result span {

    flex: 1;
```

```
padding: 10px;

border: 1px solid teal;

border-radius: 4px;}
```

4.3 JavaScript Logic

The core functionality is handled by JavaScript. When the user clicks the "Calculate" button, JavaScript reads the inputted income and applies the appropriate tax rates based on the income bracket. The tax owed is then calculated and displayed.

The tax calculation logic is divided into the following income brackets:

- **Up to \$10,000:** No tax is owed.
- **Between \$10,001 and \$40,000:** A 10% tax is applied to the income above \$10,000.
- **Between \$40,001 and \$100,000:** A 20% tax is applied to the income above \$40,000, in addition to the tax from the previous bracket.
- **Above \$100,000:** A 30% tax is applied to the income above \$100,000, plus the taxes from previous brackets.

```
•
• // function to calculate income tax owed
• let incomeCalculator = ()=>{
•   //initializing local variables
•   let income =
parseFloat(document.getElementById('income').value); //typecasting
string to float for calculations and storing it in 'income'
variable. Literals gives Not a Number (NaN) output
•   let tax = 0; //initializing variable named 'tax'
•
•
•
•   //conditional statements to apply different tax rates based on
income brackets
•   if (income < 0) {
•       alert("Please input a figure greater than zero")
•   }
```

```

    //messagebox pops up if income is negative
    }

    else (income <= 10000) {
    if
        tax = 0;
    }

    else (income <= 40000) {
    if
        tax = (income-10000)*.10;
    }

    else (income <= 100000) {
    if
        tax = (income - 10000)*.10 + (40000 - 10000)*.10
        + (100000 - 40000)*.20
    }

    else{
        tax = (income-100000)*.30 + (100000 - 40000)*.20 + (40000 -
10000)*.10
    }

    return tax.toFixed(2) //rounding tax figure to 2 decimals
}

let btn = document.getElementById('btn');

//event listener to call incomeCalculator function and update
taxOwed variable

btn.addEventListener("click", ()=> {

    let taxOwed = incomeCalculator(); //storing and updating

```

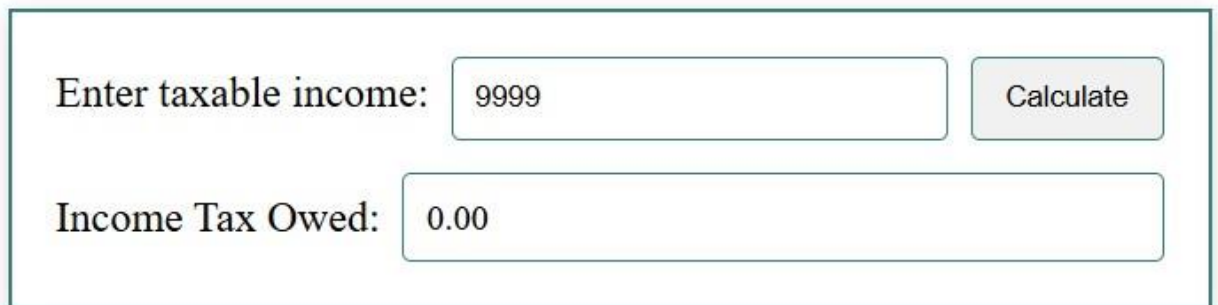
```
function output in variable 'taxOwed'
```

```
• document.getElementById('show').innerText = taxOwed; //updating  
  tax owed figure  
• })
```

6. Testing and Validation

The application was tested with various income inputs to ensure the correct tax amounts were calculated based on the specified tax brackets. Test cases included:

- Income below \$10,000: \$0 tax.



The screenshot shows a web form with two rows. The first row is labeled 'Enter taxable income:' and contains a text input field with the value '9999' and a 'Calculate' button to its right. The second row is labeled 'Income Tax Owed:' and contains a text input field with the value '0.00'.

Figure 2 Tax when income is 9999

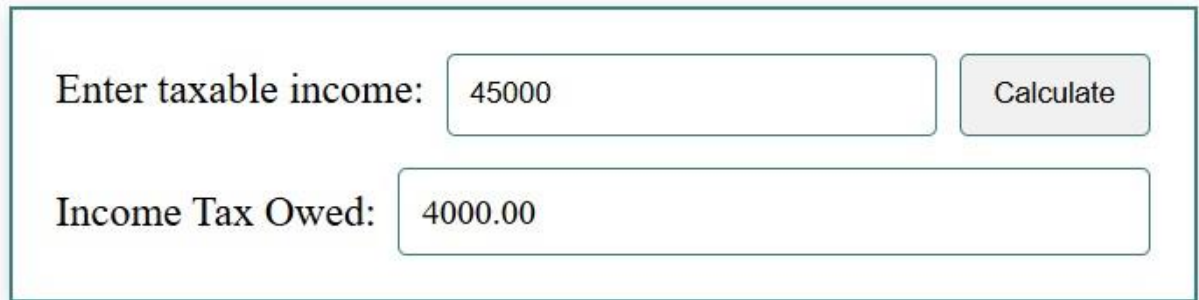
- Income between \$10,001 and \$40,000: Correct 10% tax applied.



The screenshot shows a web form with two rows. The first row is labeled 'Enter taxable income:' and contains a text input field with the value '11000' and a 'Calculate' button to its right. The second row is labeled 'Income Tax Owed:' and contains a text input field with the value '100.00'.

Figure 3 Tax when income is 1100

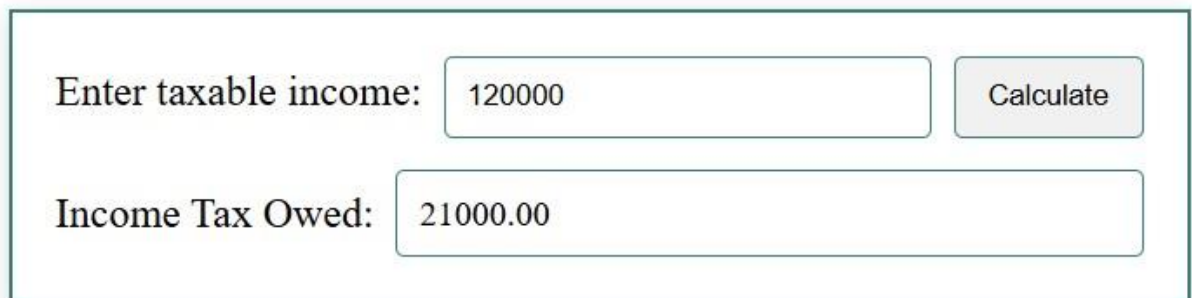
- Income between \$40,001 and \$100,000: Correct 10% and 20% tax combination applied.



Enter taxable income:	<input type="text" value="45000"/>	<input type="button" value="Calculate"/>
Income Tax Owed:	<input type="text" value="4000.00"/>	

Figure 4 Tax figure when Income is 45000

- Income above \$100,000: Correct 10%, 20%, and 30% tax combination applied.



Enter taxable income:	<input type="text" value="120000"/>	<input type="button" value="Calculate"/>
Income Tax Owed:	<input type="text" value="21000.00"/>	

Figure 5 Tax figure when Income is 120000

7. Conclusion

The Income Tax Calculator project successfully implements a user-friendly interface for calculating taxes based on different income levels. The use of clear and intuitive design elements combined with efficient JavaScript logic ensures that users can compute their tax obligations with ease. The project demonstrates the effective use of web technologies like HTML, CSS, and JavaScript to create a functional and visually appealing tax calculator.