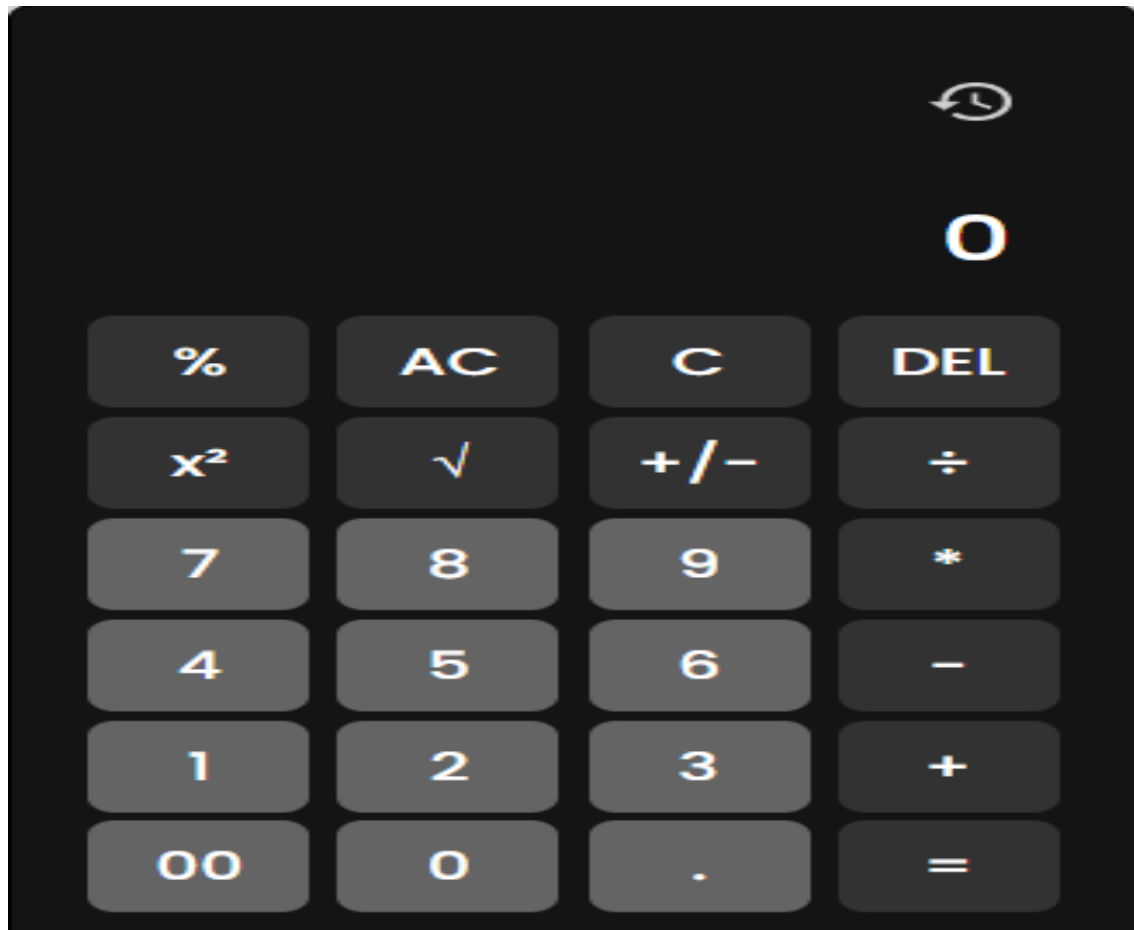


MIAGE Casablanca

## RAPPORT DE PROJET

# THEME

## STANDARD CALCULATRICE AVEC HISTORIQUE



Réalisé Par :

- ABDELILAH ZLAYJI

# **RÉALISATION D'UNE CALCULATRICE SIMPLE**

OBJECTIF .....	2
PRESENTATION DES CLASSES UTILISE EN JS .....	3
LE CODE EN HTML .....	3
LE CODE EN CSS .....	6
LE CODE EN JAVASCRIPT .....	9
INTERFACE SIMPLE (TEST) .....	14

## **TABLE DES FIGURES**












Figure I Exemple de multiplication de deux nombres .....	14
Figure II exemple de MOD de deux nombres .....	14
Figure III bouton All Clear pour effacer l'écran d'affichage .....	15

## **OBJECTIF**

**L'objectif de ce projet est de fournir une interface pour effectuer des calculs mathématiques de base tout en conservant un historique des calculs précédents.**






**Cela permet à l'utilisateur de suivre les étapes de calcul et de référencer les résultats précédents si nécessaire.**

# PRÉSENTATION DES CLASSES UTILISÉ EN JS


-  **DISPLAYHISTORY** : Cette fonction gère l'affichage ou la non-affichage de l'historique des calculs.
-  **ADDTOHISTORY** : Ajoute une entrée à l'historique des calculs.
-  **DISPLAYNUMBER** : Affiche le chiffre correspondant à celui cliqué.
-  **DISPLAYDOUBLEZERO** : Affiche "00" s'il est cliqué lorsque l'entrée précédente n'est pas un zéro.
-  **CLEARINPUT** : Efface l'écran, réinitialise les opérandes et l'opérateur.
-  **DELETEINPUT** : Supprime le dernier caractère de l'écran.
-  **CHANGESIGN** : Change le signe du nombre affiché.
-  **PERFORMCALCULATION** : Effectue le calcul en fonction de l'opérande, de l'opérateur et de l'entrée actuelle.
-  **SQUAREROOTOF** : Calcule la racine carrée du nombre affiché.
-  **SQUAREOF** : Calcule le carré du nombre affiché.
-  **RESTARTALL** : Recharge la page pour tout réinitialiser.


## LE CODE EN HTML

### 1. Classes utilisées dans les balises <div> :

-  **main**: Utilisée pour définir la section principale de la page.
-  **calculator-container**: Contient la calculatrice et son interface.
-  **display-row**: Utilisée pour la rangée de l'affichage de la calculatrice.
-  **button-row**: Utilisée pour chaque rangée de boutons de la calculatrice.
-  **history**: Contient l'historique des calculs.

### 2. Classes utilisées dans les balises <p>:

-  **equation**: Utilisée pour afficher l'équation en cours.

 **inputAnswer**: Utilisée pour afficher le résultat de l'équation en cours.

### 3. Classes utilisées dans les balises <button>

 **operator**: Utilisée pour les boutons d'opérateurs mathématiques.  
 **number**: Utilisée pour les boutons de chiffres.

### 4. Autres classes:

 **showHistory**: Utilisée pour le lien d'affichage de l'historique.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Standard Calculator with History</title>

  <!-- Style CSS -->
  <link rel="stylesheet" href="./style.css">
</head>
<body>
  <h1>-Standard Calculator with History-</h1>

  <div class="main">

    <div class="calculator-container">
      <div class="display-row">
        <a id="showHistory"></a>
        <p class="equation" id="equation">0</p>
        <p class="inputAnswer" id="inputAnswer">0</p>
      </div>
      <div class="button-row">
        <button class="operator">%</button>
        <button id="allClear">AC</button>
        <button id="clear">C</button>
```

```

    <button id="delete">DEL</button>
  </div>
  <div class="button-row">
    <button id="squared">x2</button>
    <button id="squareRoot">√</button>
    <button id="positiveNegative">+/-</button>
    <button class="operator">÷</button>
  </div>
  <div class="button-row">
    <button class="number" id="seven">7</button>
    <button class="number" id="eight">8</button>
    <button class="number" id="nine">9</button>
    <button class="operator">*</button>
  </div>
  <div class="button-row">
    <button class="number" id="four">4</button>
    <button class="number" id="five">5</button>
    <button class="number" id="six">6</button>
    <button class="operator">-</button>
  </div>
  <div class="button-row">
    <button class="number" id="one">1</button>
    <button class="number" id="two">2</button>
    <button class="number" id="three">3</button>
    <button class="operator">+</button>
  </div>
  <div class="button-row">
    <button class="number" id="doubleZero">00</button>
    <button class="number" id="zero">0</button>
    <button class="number" id="decimal">.</button>
    <button id="calculate">=</button>
  </div>
</div>
<div class="history">
  <p>History:</p>
  <ul>
  </ul>
</div>
</div>

```

```
<!-- Script JS -->
<script src="./script.js"></script>
</body>
</html>
```

## LE CODE EN CSS

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght
@500&display=swap');

* {
  margin: 0;
  padding: 0;
  font-family: 'Poppins', sans-serif;
}

body {
  background-color: rgb(210, 210, 210);
  display: flex;
  justify-content: center;
  align-items: center;
  flex-direction: column;
}

h1 {
  font-size: 40px;
  margin-top: 100px;
}

.main {
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  flex-direction: row;
  margin-top: 90px;
  max-width: 900px;
```

```

}

.calculator-container {
  flex: 1;
  box-sizing: border-box;
  border: 2px solid;
  padding: 25px;
  border-radius: 10px;
  background-color: rgb(20, 20, 20);
}

.calculator-container button {
  width: 80px;
  height: 60px;
  border-radius: 10px;
  font-size: 25px;
  margin: 3px;
  color: rgb(255, 255, 255);
  background-color: rgb(50, 50, 50);
  border: none;
  cursor: pointer;
}

.calculator-container button:hover {
  background-color: rgb(90, 90, 90);
}

.display-row {
  height: 130px;
  max-height: 130px;
  text-align: right;
  color: rgb(255, 255, 255);
  border-radius: 10px;
  padding: 15px;
  margin: 10px 5px 5px 5px;
}

.display-row .equation {
  visibility: hidden;
  font-size: 22px;
  margin-top: 40px;
}

.display-row .inputAnswer {

```

```

        font-size: 40px;
    }

    .calculator-container .number {
        background-color: rgb(100, 100, 100);
    }

    .history {
        box-sizing: border-box;
        background-color: rgb(20, 20, 20);
        color: rgb(255, 255, 255);
        font-size: 25px;
        text-align: center;
        width: 230px;
        height: 623px;
        border-radius: 0px 20px 20px 0px;
    }

    .history ul {
        list-style: none;
        margin-top: 20px;
        overflow-y: auto;
        max-height: 540px;
        width: 210px;
    }

    .history p {
        margin-top: 20px;
        text-align: left;
        margin-left: 15px;
        font-size: 18px;
    }
    a img {
        height: 25px;
        width: 30px;
        float: right;
    }

```



# LE CODE EN JAVASCRIPT

```
const numbers = document.querySelectorAll('.number');
const operators = document.querySelectorAll('.operator');
const inputAnswer = document.querySelector('.inputAnswer');
const equation = document.querySelector('.equation');
const clear = document.getElementById('clear');
const deleteInputs = document.getElementById('delete');
const historyArea = document.querySelector('.history');
const mainArea = document.querySelector('.calculator-
container');

historyArea.style.display = "none";

function displayHistory() {
    if (historyArea.style.display === "none") {
        historyArea.style.display = "";
        mainArea.style.borderRadius = "10px 0px 0px 10px";
    } else {
        historyArea.style.display = "none";
        mainArea.style.marginLeft = "";
        mainArea.style.borderRadius = "10px";
    }
}

document.getElementById('showHistory').addEventListener('click'
, displayHistory);

const history = [];

function addToHistory(entry) {
    history.push(entry);

    const historyList = document.querySelector('.history ul');
    const newHistoryItem = document.createElement('li');
    newHistoryItem.textContent = entry;
    historyList.appendChild(newHistoryItem);
}

let operand1 = null;
let operator = null;

function displayNumber(event) {
    const clickedNumber = event.target;
```

```

    const inputValue = inputAnswer.textContent;

    if (inputValue === '0') {
        inputAnswer.textContent = clickedNumber.textContent;
    } else {
        inputAnswer.textContent += clickedNumber.textContent;
    }
}

numbers.forEach(number => {
    number.addEventListener('click', displayNumber);
});

function displayDoubleZero(event) {
    const clickedNumber = event.target;
    const inputValue = inputAnswer.textContent;

    if (inputValue > 0) {
        inputValue.textContent += clickedNumber.textContent;
    } else {
        inputAnswer.textContent = 0;
    }
}

document.getElementById('doubleZero').addEventListener('click',
displayDoubleZero);

function clearInput() {
    equation.style.visibility = "hidden";
    inputAnswer.textContent = '0';
    operand1 = null;
    operator = null;
    equation.textContent = '0';
}

clear.addEventListener('click', clearInput);

function deleteInput() {
    let inputValue = inputAnswer.textContent;
    if (inputValue.length > 1) {
        inputValue = inputValue.slice(0, -1);
        inputAnswer.textContent = inputValue;
    } else {
        inputAnswer.textContent = '0';
    }
}

deleteInputs.addEventListener('click', deleteInput);

```

```

function changeSign() {
    const currentText = inputAnswer.textContent;

    if (currentText !== '0') {
        if (currentText.startsWith('-')) {
            inputAnswer.textContent = currentText.slice(1);
        } else {
            inputAnswer.textContent = '-' + currentText;
        }
    }
}

document.getElementById('positiveNegative').addEventListener('click', changeSign);

function performCalculation() {
    const inputValue = inputAnswer.textContent;

    if (operand1 !== null && operator !== null) {
        let result;

        switch (operator) {
            case '+':
                result = (operand1 +
parseFloat(inputValue));
                break;
            case '-':
                result = (operand1 -
parseFloat(inputValue));
                break;
            case '*':
                result = (operand1 *
parseFloat(inputValue));
                break;
            case '÷':
                result = (operand1 /
parseFloat(inputValue));
                break;
            case '%':
                result = ((operand1 / 100) *
parseFloat(inputValue));
                break;
            default:
                break;
        }
    }
}

```

```

        if (result % 1 === 0) {
            inputAnswer.textContent = result.toFixed(0);
        } else {
            inputAnswer.textContent = result.toFixed(2);
        }

        equation.textContent = operand1 + " " + operator +
        " " + inputValue + " = ";

        const historyEntry = `${equation.textContent}
        ${inputAnswer.textContent}`;
        addToHistory(historyEntry);

        operand1 = null;
        operator = null;

        operand1 = null;
        operator = null;
    }
}

operators.forEach(operatorButton => {
    operatorButton.addEventListener('click', () => {
        equation.style.visibility = "visible";
        operand1 = parseFloat(inputAnswer.textContent);
        operator = operatorButton.textContent;
        equation.textContent = operand1 + " " + operator;
        inputAnswer.textContent = '0';
    });
})

document.getElementById('calculate').addEventListener('click',
performCalculation);

function squareRootOf() {
    const inputValue = parseFloat(inputAnswer.textContent);

    if (inputValue > 0) {
        let result;
        equation.style.visibility = "visible";
        equation.textContent = "√ " + inputValue;
        result = Math.sqrt(inputValue);

        if (result % 1 === 0) {
            inputAnswer.textContent = result.toFixed(0);
        } else {
            inputAnswer.textContent = result.toFixed(2);
        }
    }
}

```

```

    }
    const historyEntry = `${equation.textContent} =
    ${inputAnswer.textContent}`;
    addToHistory(historyEntry);
}
document.getElementById('squareRoot').addEventListener('click',
squareRootOf);

function squareOf() {
    const inputValue = parseFloat(inputAnswer.textContent);

    if (inputValue >= 0) {
        let result;
        equation.style.visibility = "visible";
        equation.textContent = inputValue + "^2";
        result = Math.pow(inputValue, 2);

        inputAnswer.textContent = result;
    }
    const historyEntry = `${equation.textContent} =
    ${inputAnswer.textContent}`;
    addToHistory(historyEntry);
}
document.getElementById('squared').addEventListener('click',
squareOf);

function restartAll(){
    location.reload();
}
document.getElementById('allClear').addEventListener('click',
restartAll);

```

## INTERFACE SIMPLE (TEST)

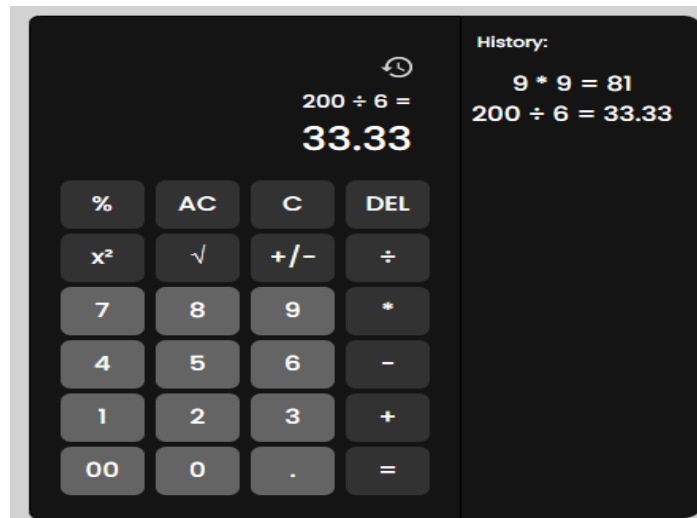


Figure I Exemple de multiplication de deux nombres

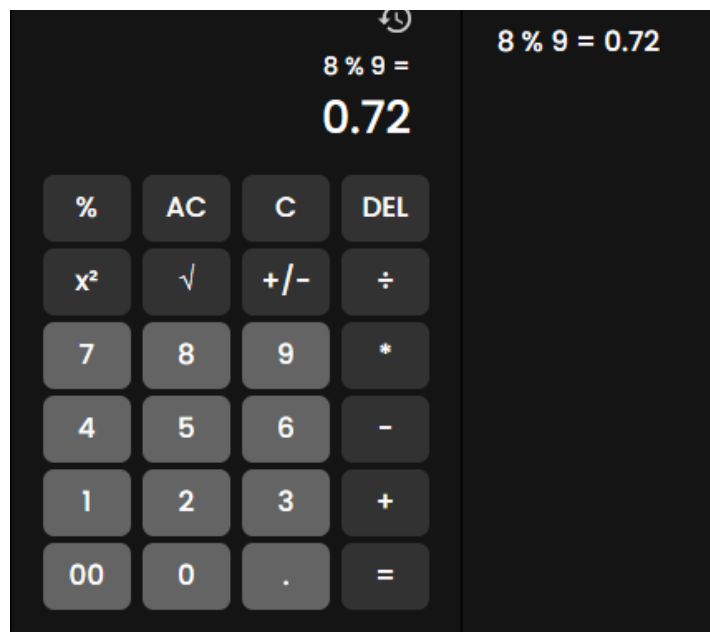


Figure II exemple de MOD de deux nombres

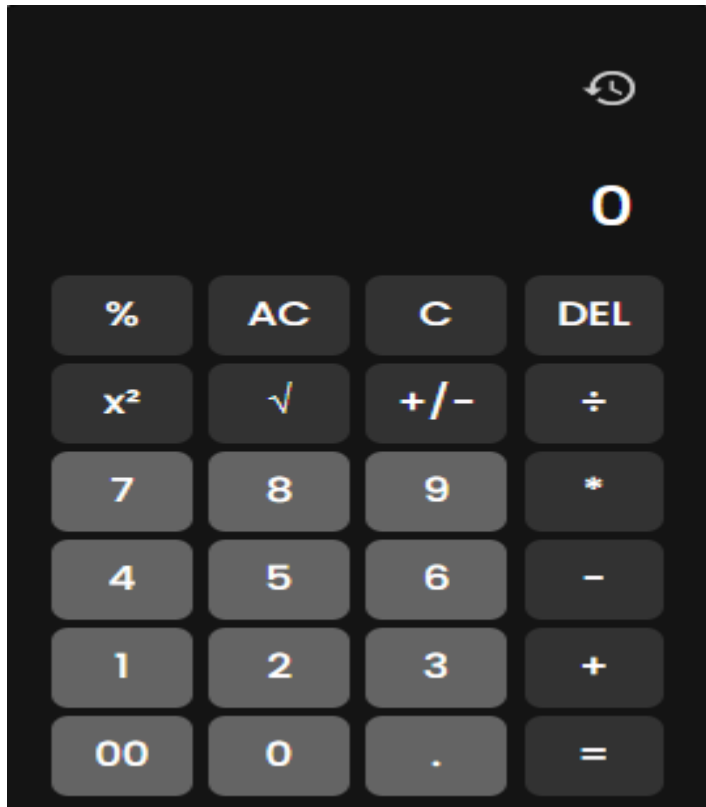
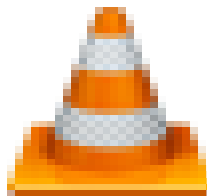


Figure III bouton All Clear pour effacer l'écran d'affichage

## **TEST VÉDIO**



Test vedio.mp4