

A screenshot of a calculator

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

HTML CODE

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

  <title>Uganda Net Pay Calculator</title>

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

  <meta name="author" content="Uganda Revenue Authority" />

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

  <!-- External Libraries -->

  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

  <script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.min.js"></script>

</head>

<body>

  <div class="container">

    <!-- Header Section -->

    <header class="header-container">

      <div class="theme-control">

        <label class="switch">

          <input type="checkbox" id="darkModeToggle" onchange="toggleDarkMode()" />

          <span class="slider round"></span>

        </label>

        <span class="toggle-label">Dark Mode</span>

      </div>

      <h1>Uganda Net Pay Calculator</h1>

      <div id="dateTime" class="date-time" aria-live="polite"></div>

    </header>

    <!-- Main Calculator Content -->

    <main>

      <!-- Input Section -->

      <section class="calculator-section">

        <form id="netPayForm">

          <div class="input-group">

            <label for="grossSalary">

              Gross Salary (UGX)

              <span class="tooltip">?

                <span class="tooltiptext">Basic salary before any deductions</span>

              </span>

            </label>

            <input

              type="number"

              id="grossSalary"

              min="0"

              required

              aria-label="Enter gross salary in Ugandan Shillings"

            />

          </div>

          <div class="input-group">

            <label for="salaryType">Salary Type:</label>

            <select id="salaryType">

              <option value="monthly" selected>Monthly</option>

              <option value="annual">Annual</option>

            </select>

          </div>

          <div class="input-group">

            <label for="nssfOption">NSSF Option:</label>

            <select id="nssfOption">

              <option value="none">None</option>

              <option value="employee">Employee Only (5%)</option>

              <option value="employer">Employer Only (10%)</option>

              <option value="both">Both Employee & Employer</option>

            </select>

          </div>

          <div class="input-group">

            <label for="housingAllowance">Housing Allowance:</label>

            <input type="number" id="housingAllowance" value="0" />

          </div>

          <div class="input-group">

            <label for="transportAllowance">Transport Allowance:</label>

            <input type="number" id="transportAllowance" value="0" />

          </div>

          <div class="input-group">

            <label for="otherDeductions">Other Deductions:</label>

            <input type="number" id="otherDeductions" value="0" />

          </div>

          <div class="action-buttons">

            <button type="button" class="primary" onclick="calculateNetPay()">

              Calculate

            </button>

            <button type="reset" class="secondary" onclick="resetForm()">Reset</button>

          </div>

        </form>

      </section>

      <!-- Results Section -->

      <section class="results-section" aria-live="polite">

        <h2>Calculation Results</h2>

        <div class="results-grid">

          <p>PAYE: <span id="payeResult">0</span> UGX</p>

          <p>NSSF (Employee): <span id="nssfEmployee">0</span> UGX</p>

          <p>NSSF (Employer): <span id="nssfEmployer">0</span> UGX</p>

          <p>Total Deductions: <span id="totalDeductions">0</span> UGX</p>

          <p class="highlight">Net Pay: <span id="netPay">0</span> UGX</p>

          <p>Employer Total Cost: <span id="employerCost">0</span> UGX</p>

        </div>

        <div class="paye-breakdown">

          <h3>PAYE Calculation Breakdown</h3>

          <pre id="payeCalculationSteps" class="breakdown-content"></pre>

        </div>

      </section>

      <!-- Visualization Section -->

      <section class="visualization">

        <div class="chart-container">

          <canvas id="deductionChart" width="400" height="200"></canvas>

        </div>

        <button onclick="downloadSummaryPDF()" class="download-btn">Download Summary</button>

      </section>

      <!-- History Section -->

      <section class="history-section" aria-label="Calculation history">

        <h2>

          <button

            class="history-toggle"

            onclick="toggleHistory()"

            aria-expanded="false"

            aria-controls="historyContent"

          >

            ▼ Calculation History

          </button>

        </h2>

        <div id="historyContent" class="history-content" hidden>

          <button onclick="clearHistory()" class="clear-history">Clear History</button>

          <ul id="historyList" role="list"></ul>

        </div>

      </section>

    </main>

    <!-- Footer Section -->

    <footer class="page-footer">

      <p>&copy; 2025 Uganda Revenue Authority &amp; SwaibuTech. All rights reserved.</p>

    </footer>

  </div>

  <script src="script.js"></script>

</body>

</html>

CSS

/\* ========== Reset & Base ========== \*/

\* {

  box-sizing: border-box;

  margin: 0;

  padding: 0;

}

body {

  font-family: 'Segoe UI', sans-serif;

  background-color: #f2f4f7;

  color: #333;

  padding: 20px;

}

.container {

  max-width: 600px;

  margin: auto;

  background: #fff;

  padding: 30px;

  border-radius: 15px;

  box-shadow: 0 8px 24px rgba(0, 0, 0, 0.1);

}

/\* ========== Headings & Header ========== \*/

h1, h2 {

  text-align: center;

  margin-bottom: 20px;

  color: #004085;

}

.header-container {

  display: flex;

  align-items: center;

  justify-content: flex-start;

  gap: 15px;

  margin-bottom: 20px;

  position: relative;

  padding: 1rem;

}

.date-time {

  position: absolute;

  top: 10px;

  right: 10px;

  font-size: 0.9rem;

  color: var(--text-color, #333);

  font-weight: 500;

  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

  user-select: none;

}

/\* ========== Form Elements ========== \*/

form {

  display: flex;

  flex-direction: column;

  gap: 15px;

}

.input-group {

  display: flex;

  justify-content: space-between;

  align-items: center;

}

.input-group label {

  flex: 1;

}

.input-group input,

.input-group select {

  flex: 2;

  padding: 8px;

  border: 1px solid #ccc;

  border-radius: 6px;

  margin-left: 10px;

}

select {

  padding: 8px;

  border-radius: 6px;

  border: 1px solid #ccc;

}

/\* ========== Buttons & Interactions ========== \*/

.button-group {

  display: flex;

  justify-content: space-between;

  margin-top: 20px;

}

button {

  padding: 10px 20px;

  border: none;

  border-radius: 8px;

  background-color: #004085;

  color: white;

  font-size: 16px;

  cursor: pointer;

  transition: background 0.3s ease;

}

button:hover {

  background-color: #0056b3;

}

.download-btn {

  margin-top: 20px;

  width: 100%;

}

/\* ========== Results Display ========== \*/

#results {

  background-color: #e2edff;

  padding: 20px;

  margin-top: 30px;

  border-radius: 12px;

  box-shadow: inset 0 0 5px rgba(0, 0, 0, 0.05);

}

#results h3 {

  color: green;

  font-size: 24px;

  margin-top: 20px;

}

/\* ========== Dark Mode Overrides ========== \*/

body.dark {

  background-color: #121212;

  color: #f1f1f1;

}

body.dark .container {

  background-color: #2c2c2c;

}

body.dark input,

body.dark select {

  background-color: #333;

  color: white;

  border: 1px solid #666;

}

body.dark button {

  background-color: #007bff;

}

body.dark .date-time {

  color: #ccc;

}

/\* ========== Toggle Switch Component ========== \*/

.toggle-wrapper {

  display: flex;

  align-items: center;

  gap: 5px;

}

.switch {

  position: relative;

  display: inline-block;

  width: 40px;

  height: 22px;

}

.switch input {

  opacity: 0;

  width: 0;

  height: 0;

}

.slider {

  position: absolute;

  cursor: pointer;

  background-color: #ccc;

  border-radius: 34px;

  top: 0;

  left: 0;

  right: 0;

  bottom: 0;

  transition: .4s;

}

.slider:before {

  position: absolute;

  content: "";

  height: 16px;

  width: 16px;

  left: 3px;

  bottom: 3px;

  background-color: white;

  transition: .4s;

  border-radius: 50%;

}

input:checked + .slider {

  background-color: #2196F3;

}

input:checked + .slider:before {

  transform: translateX(18px);

}

/\* ========== Toast Component ========== \*/

.toast {

  position: fixed;

  bottom: 20px;

  right: 20px;

  padding: 12px 20px;

  border-radius: 8px;

  color: white;

  z-index: 1000;

  animation: slideIn 0.3s ease-out;

}

.toast.error { background: #dc3545; }

.toast.info { background: #17a2b8; }

@keyframes slideIn {

  from { transform: translateX(100%); }

  to { transform: translateX(0); }

}

/\* ========== Tooltip Component ========== \*/

.tooltip {

  position: relative;

  display: inline-block;

  cursor: help;

  margin-left: 5px;

}

.tooltiptext {

  visibility: hidden;

  width: 200px;

  background: #333;

  color: #fff;

  text-align: center;

  border-radius: 6px;

  padding: 5px;

  position: absolute;

  z-index: 1;

  bottom: 125%;

  left: 50%;

  transform: translateX(-50%);

  opacity: 0;

  transition: opacity 0.3s;

}

.tooltip:hover .tooltiptext {

  visibility: visible;

  opacity: 1;

}

/\* ========== Accessibility ========== \*/

input:focus,

select:focus {

  outline: 3px solid #0056b3;

  outline-offset: 2px;

}

/\* Add to your CSS \*/

.history-section {

    margin-top: 30px;

    border: 1px solid #ddd;

    border-radius: 8px;

    padding: 15px;

}

.history-content {

    display: none;

    margin-top: 10px;

}

.history-content.show {

    display: block;

}

#historyList {

    list-style: none;

    padding: 0;

    max-height: 300px;

    overflow-y: auto;

}

#historyList li {

    background: #f8f9fa;

    padding: 10px;

    margin: 5px 0;

    border-radius: 5px;

    border: 1px solid #eee;

    display: flex;

    justify-content: space-between;

    align-items: center;

}

.history-details {

    font-size: 0.9em;

    color: #666;

}

.clear-history {

    background: #dc3545;

    margin-bottom: 10px;

}

.clear-history:hover {

    background: #bb2d3b;

}

JS CODE

// ====================== GLOBAL VARIABLES ======================

let chart; // Chart.js instance

// ====================== INITIALIZATION ======================

window.addEventListener('DOMContentLoaded', () => {

    loadFormState();

    displayHistory();

    updateDateTime();

    setInterval(updateDateTime, 1000);

});

// ====================== EVENT HANDLERS ======================

document.querySelectorAll('input, select').forEach(element => {

    element.addEventListener('input', () => validateInput(element));

    element.addEventListener('change', saveFormState);

});

// ====================== CORE CALCULATIONS ======================

function calculateNetPay() {

    // Inputs

    const grossInput = parseFloat(document.getElementById("grossSalary").value) || 0;

    const salaryType = document.getElementById("salaryType").value;

    const gross = salaryType === "annual" ? grossInput / 12 : grossInput;

    const housing = parseFloat(document.getElementById("housingAllowance").value) || 0;

    const transport = parseFloat(document.getElementById("transportAllowance").value) || 0;

    const otherDeductions = parseFloat(document.getElementById("otherDeductions").value) || 0;

    const totalAllowances = housing + transport;

    const taxableIncome = gross + totalAllowances;

    // Deductions

    const paye = calculatePAYE(taxableIncome);

    const { employee: nssfEmp, employer: nssfEr } = calculateNSSF(gross);

    const totalDeductions = paye + nssfEmp + otherDeductions;

    const netPay = gross + totalAllowances - totalDeductions;

    // Display

    updateDisplay(paye, nssfEmp, nssfEr, totalDeductions, netPay);

    updateChart(paye, nssfEmp, otherDeductions);

    addToHistory();

}

// PAYE Calculation Logic (Uganda)

function calculatePAYE(taxableIncome) {

    if (taxableIncome <= 235000) return 0;

    if (taxableIncome <= 335000) return 0.1 \* (taxableIncome - 235000);

    if (taxableIncome <= 410000) return 10000 + 0.2 \* (taxableIncome - 335000);

    return 25000 + 0.3 \* (taxableIncome - 410000);

}

// NSSF Calculation

function calculateNSSF(gross) {

    const NSSF\_CAP = 240000;

    const cappedSalary = Math.min(gross, NSSF\_CAP);

    const option = document.getElementById("nssfOption").value;

    return {

        employee: {

            employee: cappedSalary \* 0.05,

            both: cappedSalary \* 0.05

        }[option] || 0,

        employer: {

            employer: cappedSalary \* 0.10,

            both: cappedSalary \* 0.10

        }[option] || 0

    };

}

// ====================== DISPLAY UPDATES ======================

function updateDisplay(paye, nssfEmp, nssfEr, totalDeductions, netPay) {

    document.getElementById("payeResult").innerText = paye.toFixed(2);

    document.getElementById("nssfEmployee").innerText = nssfEmp.toFixed(2);

    document.getElementById("nssfEmployer").innerText = nssfEr.toFixed(2);

    document.getElementById("totalDeductions").innerText = totalDeductions.toFixed(2);

    document.getElementById("netPay").innerText = netPay.toFixed(2);

    const gross = parseFloat(document.getElementById("grossSalary").value) || 0;

    const housing = parseFloat(document.getElementById("housingAllowance").value) || 0;

    const transport = parseFloat(document.getElementById("transportAllowance").value) || 0;

    explainPAYE(gross + housing + transport);

}

function explainPAYE(taxableIncome) {

    const format = num => num.toLocaleString("en-UG");

    let steps = "";

    if (taxableIncome > 410000) {

        steps = `• First 410,000: 25,000 UGX\n• Remaining ${format(taxableIncome - 410000)} @30%: UGX ${(0.3 \* (taxableIncome - 410000)).toFixed(2)}`;

    } else if (taxableIncome > 335000) {

        steps = `• First 335,000: 10,000 UGX\n• Remaining ${format(taxableIncome - 335000)} @20%: UGX ${(0.2 \* (taxableIncome - 335000)).toFixed(2)}`;

    } else if (taxableIncome > 235000) {

        steps = `• First 235,000: 0 UGX\n• Remaining ${format(taxableIncome - 235000)} @10%: UGX ${(0.1 \* (taxableIncome - 235000)).toFixed(2)}`;

    }

    document.getElementById("payeCalculationSteps").textContent = steps;

}

// ====================== CHART UPDATES ======================

function updateChart(paye, nssf, others) {

    if (chart) chart.destroy();

    chart = new Chart(document.getElementById('deductionChart').getContext('2d'), {

        type: 'pie',

        data: {

            labels: ['PAYE', 'NSSF (Employee)', 'Other Deductions'],

            datasets: [{

                data: [paye, nssf, others],

                backgroundColor: ['#ff6384', '#36a2eb', '#ffcd56']

            }]

        },

        options: {

            responsive: true,

            plugins: {

                legend: { position: 'top' },

                title: { display: true, text: 'Deduction Breakdown' }

            }

        }

    });

}

function downloadSummaryPDF() {

  const { jsPDF } = window.jspdf;

  const netPay = document.getElementById("netPay").textContent;

  const paye = document.getElementById("payeResult").textContent;

  const nssfEmployee = document.getElementById("nssfEmployee").textContent;

  const nssfEmployer = document.getElementById("nssfEmployer").textContent;

  const totalDeductions = document.getElementById("totalDeductions").textContent;

  const employerCost = document.getElementById("employerCost").textContent;

  const breakdown = document.getElementById("payeCalculationSteps").textContent;

  const doc = new jsPDF();

  let y = 10; // vertical starting point

  doc.setFontSize(16);

  doc.text("Uganda Net Pay Summary", 10, y);

  y += 10;

  doc.setFontSize(12);

  doc.text(`Net Pay: ${netPay} UGX`, 10, y);

  y += 7;

  doc.text(`PAYE: ${paye} UGX`, 10, y);

  y += 7;

  doc.text(`NSSF (Employee): ${nssfEmployee} UGX`, 10, y);

  y += 7;

  doc.text(`NSSF (Employer): ${nssfEmployer} UGX`, 10, y);

  y += 7;

  doc.text(`Total Deductions: ${totalDeductions} UGX`, 10, y);

  y += 7;

  doc.text(`Employer Total Cost: ${employerCost} UGX`, 10, y);

  y += 10;

  doc.text("PAYE Breakdown:", 10, y);

  y += 7;

  // Add breakdown text, split into lines to avoid overflow

  const splitText = doc.splitTextToSize(breakdown, 180);

  doc.text(splitText, 10, y);

  doc.save("net\_pay\_summary.pdf");

}

// ====================== HISTORY HANDLING ======================

function addToHistory() {

    const history = JSON.parse(localStorage.getItem('salaryHistory') || '[]');

    const entry = {

        timestamp: new Date().toISOString(),

        gross: document.getElementById('grossSalary').value,

        salaryType: document.getElementById('salaryType').value,

        paye: document.getElementById('payeResult').innerText,

        nssfEmp: document.getElementById('nssfEmployee').innerText,

        nssfEr: document.getElementById('nssfEmployer').innerText,

        netPay: document.getElementById('netPay').innerText,

        allowances: {

            housing: document.getElementById('housingAllowance').value,

            transport: document.getElementById('transportAllowance').value

        },

        deductions: document.getElementById('otherDeductions').value

    };

    if (history.length >= 10) history.shift();

    history.push(entry);

    localStorage.setItem('salaryHistory', JSON.stringify(history));

    displayHistory();

}

function displayHistory() {

    const history = JSON.parse(localStorage.getItem('salaryHistory') || '[]');

    const list = document.getElementById('historyList');

    list.innerHTML = '';

    history.reverse().forEach(entry => {

        const li = document.createElement('li');

        li.innerHTML = `

            <div>

                <strong>${new Date(entry.timestamp).toLocaleString()}</strong>

                <div class="history-details">

                    Net Pay: UGX ${parseFloat(entry.netPay).toLocaleString('en-UG')} |

                    Gross: UGX ${parseFloat(entry.gross).toLocaleString('en-UG')} (${entry.salaryType})

                </div>

            </div>

            <div>

                <button onclick="loadHistoryEntry('${entry.timestamp}')">View</button>

            </div>

        `;

        list.appendChild(li);

    });

}

function loadHistoryEntry(timestamp) {

    const history = JSON.parse(localStorage.getItem('salaryHistory') || '[]');

    const entry = history.find(e => e.timestamp === timestamp);

    if (!entry) return;

    document.getElementById('grossSalary').value = entry.gross;

    document.getElementById('salaryType').value = entry.salaryType;

    document.getElementById('housingAllowance').value = entry.allowances.housing;

    document.getElementById('transportAllowance').value = entry.allowances.transport;

    document.getElementById('otherDeductions').value = entry.deductions;

    calculateNetPay();

}

function clearHistory() {

    localStorage.removeItem('salaryHistory');

    displayHistory();

}

// ====================== FORM STATE ======================

function saveFormState() {

    const formData = {

        grossSalary: document.getElementById("grossSalary").value,

        nssfOption: document.getElementById("nssfOption").value,

        housingAllowance: document.getElementById("housingAllowance").value,

        transportAllowance: document.getElementById("transportAllowance").value,

        otherDeductions: document.getElementById("otherDeductions").value,

        salaryType: document.getElementById("salaryType").value

    };

    localStorage.setItem("salaryCalculatorData", JSON.stringify(formData));

}

function loadFormState() {

    const savedData = JSON.parse(localStorage.getItem("salaryCalculatorData"));

    if (!savedData) return;

    document.getElementById("grossSalary").value = savedData.grossSalary;

    document.getElementById("nssfOption").value = savedData.nssfOption;

    document.getElementById("housingAllowance").value = savedData.housingAllowance;

    document.getElementById("transportAllowance").value = savedData.transportAllowance;

    document.getElementById("otherDeductions").value = savedData.otherDeductions;

    document.getElementById("salaryType").value = savedData.salaryType || "monthly";

}

// ====================== UTILITIES ======================

function validateInput(input) {

    if (input.value < 0) {

        input.value = 0;

        showToast("Negative values not allowed!", "error");

    }

}

function showToast(message, type = "info") {

    const toast = document.createElement("div");

    toast.className = `toast ${type}`;

    toast.textContent = message;

    document.body.appendChild(toast);

    setTimeout(() => toast.remove(), 3000);

}

function updateDateTime() {

    const options = {

        weekday: 'long',

        year: 'numeric',

        month: 'long',

        day: 'numeric',

        hour: '2-digit',

        minute: '2-digit',

        second: '2-digit',

        hour12: false

    };

    document.getElementById('dateTime').textContent = new Date().toLocaleString('en-GB', options);

}

// ====================== UI CONTROLS ======================

function resetForm() {

    document.getElementById("netPayForm").reset();

    document.querySelectorAll('[id$="Result"], #nssfEmployee, #nssfEmployer, #totalDeductions, #netPay')

        .forEach(el => el.innerText = "0");

    if (chart) chart.destroy();

}

function toggleDarkMode() {

    document.body.classList.toggle('dark');

}

function toggleHistory() {

    const content = document.getElementById('historyContent');

    content.classList.toggle('show');

    document.querySelector('.history-section h2').textContent =

        content.classList.contains('show') ? '▼ Calculation History' : '▶ Calculation History';

}