**Dynamic Inventory Management System: Design an inventory management system for a warehouse using a 1-D array. The array should store product details (ID, name, quantity, and price). Implement operations for adding new products, updating inventory levels, and calculating the total inventory value dynamically.**

<html>

<head>

<title>Inventory Management System</title>

<style>

body {

font-family: Arial, sans-serif;

}

table {

width: 100%;

border-collapse: collapse;

margin-top: 20px;

}

th, td {

border: 1px solid #000;

padding: 8px;

text-align: left;

}

input {

margin: 5px;

}

</style>

</head>

<body>

<h1>Inventory Management System</h1>

<h2>Add New Product</h2>

<form id="addProductForm">

<input type="text" id="productId" placeholder="Product ID" required>

<input type="text" id="productName" placeholder="Product Name" required>

<input type="number" id="productQuantity" placeholder="Quantity" required>

<input type="number" id="productPrice" placeholder="Price" required>

<button type="submit">Add Product</button>

</form>

<h2>Update Product Quantity</h2>

<form id="updateProductForm">

<input type="text" id="updateProductId" placeholder="Product ID" required>

<input type="number" id="updateQuantity" placeholder="New Quantity" required>

<button type="submit">Update Quantity</button>

</form>

<h2>Inventory</h2>

<table id="inventoryTable">

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Quantity</th>

<th>Price</th>

</tr>

</thead>

<tbody id="inventoryBody"></tbody>

</table>

<h2>Total Inventory Value: Rs. <span id="totalValue"></span></h2>

<script>

const inventory = []; // 1D array to store product details

// Function to add a new product

function addProduct(id, name, quantity, price) {

const product = { id, name, quantity, price };

inventory.push(product);

updateInventoryDisplay();

}

// Function to update product quantity

function updateProductQuantity(id, newQuantity) {

for (let i = 0; i < inventory.length; i++) {

if (inventory[i].id === id) {

inventory[i].quantity = newQuantity;

updateInventoryDisplay();

return; // Exit the function after updating

}

}

alert('Product not found!');

}

// Function to calculate total inventory value

function calculateTotalValue() {

let total = 0;

for (let i = 0; i < inventory.length; i++) {

total += inventory[i].quantity \* inventory[i].price;

}

return total;

}

// Function to update the display of the inventory

function updateInventoryDisplay() {

const inventoryBody = document.getElementById('inventoryBody');

inventoryBody.innerHTML = ''; // Clear existing entries

inventory.forEach(product => {

const row = document.createElement('tr');

row.innerHTML = `

<td>${product.id}</td>

<td>${product.name}</td>

<td>${product.quantity}</td>

<td>${product.price}</td>

`;

inventoryBody.appendChild(row);

});

const totalValue = calculateTotalValue();

document.getElementById('totalValue').innerText = totalValue;

}

// Event listeners for forms

document.getElementById('addProductForm').addEventListener('submit', function(event) {

event.preventDefault();

const id = document.getElementById('productId').value;

const name = document.getElementById('productName').value;

const quantity = parseInt(document.getElementById('productQuantity').value);

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

addProduct(id, name, quantity, price);

this.reset(); // Clear the form

});

document.getElementById('updateProductForm').addEventListener('submit', function(event) {

event.preventDefault();

const id = document.getElementById('updateProductId').value;

const newQuantity = parseInt(document.getElementById('updateQuantity').value);

updateProductQuantity(id, newQuantity);

this.reset(); // Clear the form

});

</script>

</body>

</html>