script.js

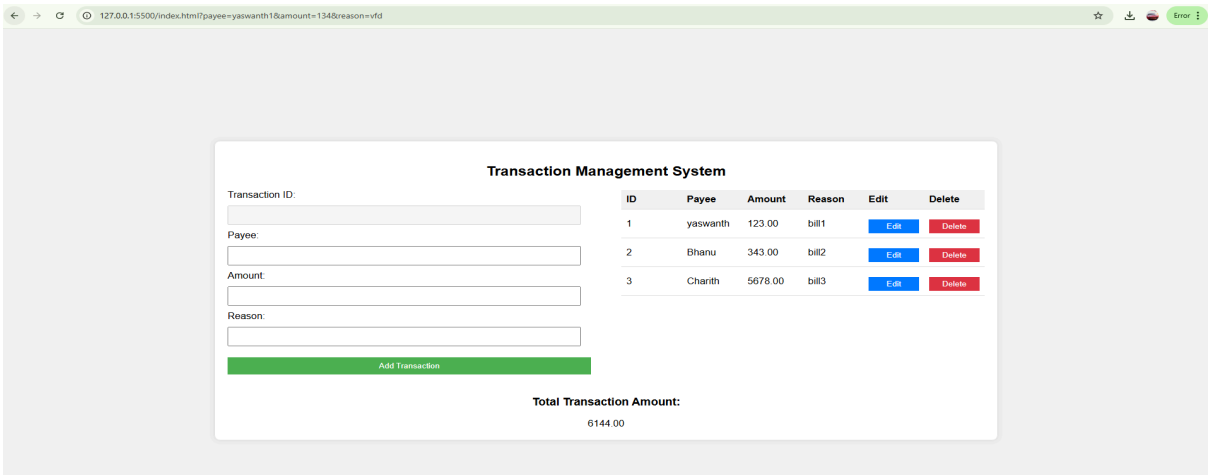
```
// task1: Complete the below statement to initialize the variable
'transactions' as an empty array.
let transactions = [];

let isEditing = false;
let editingTransactionId = null;


document.getElementById('transactionForm').addEventListener('submit',
async function (event) {
    event.preventDefault(); // Prevent default form submission


    // task2: Retrieve 'payee', 'amount', and 'reason' values from form
inputs using document.getElementById().

    // Replace '' with code to access 'payee' input value from HTML
document
    const payee = document.getElementById("payee").value;

    // Replace '' with code to access 'amount' input value from HTML
document and convert it to float
    const amount = parseFloat(document.getElementById("amount").value);

    // Replace '' with code to access 'reason' input value from HTML
document
    const reason = document.getElementById("reason").value;


    if (payee === '' || amount <= 0 || isNaN(amount)) {
        alert('Please enter valid transaction details or check your
script code.');
        return;
    }

    if (isEditing) {
        updateTransaction(editingTransactionId, payee, amount, reason);
        isEditing = false;
        editingTransactionId = null;
        document.querySelector('button[type="submit"]').textContent =
'Add Transaction';
```

```
        await updateTotalTransactionAmount();
    } else {
        addTransaction(payee, amount, reason);
    }

    clearForm();
    updateTransactionTable();
    await updateTotalTransactionAmount();
});


function addTransaction(payee, amount, reason) {
    // Note: Generated a unique 'id' for each new transaction
    const id = transactions.length > 0 ?
transactions[transactions.length - 1].id + 1 : 1;

    // task3: Complete the 'newTransaction' object by replacing the
placeholder values with appropriate data.
    const newTransaction = {
        id: id,
        payee: payee,
        amount: amount,
        reason: reason
    };


    // task4: Complete the below statement to add the 'newTransaction'
object into the 'transactions' array using the 'push' method.
    transactions.push(newTransaction);

}

function updateTransaction(id, payee, amount, reason) {
    // task5: Complete the below statement to find the transaction
object in the 'transactions' array that matches the provided 'id'
parameter.
    const transaction = transactions.find(t => t.id === id);


    if (transaction) {
        transaction.payee = payee;
        transaction.amount = amount;
        transaction.reason = reason;
```

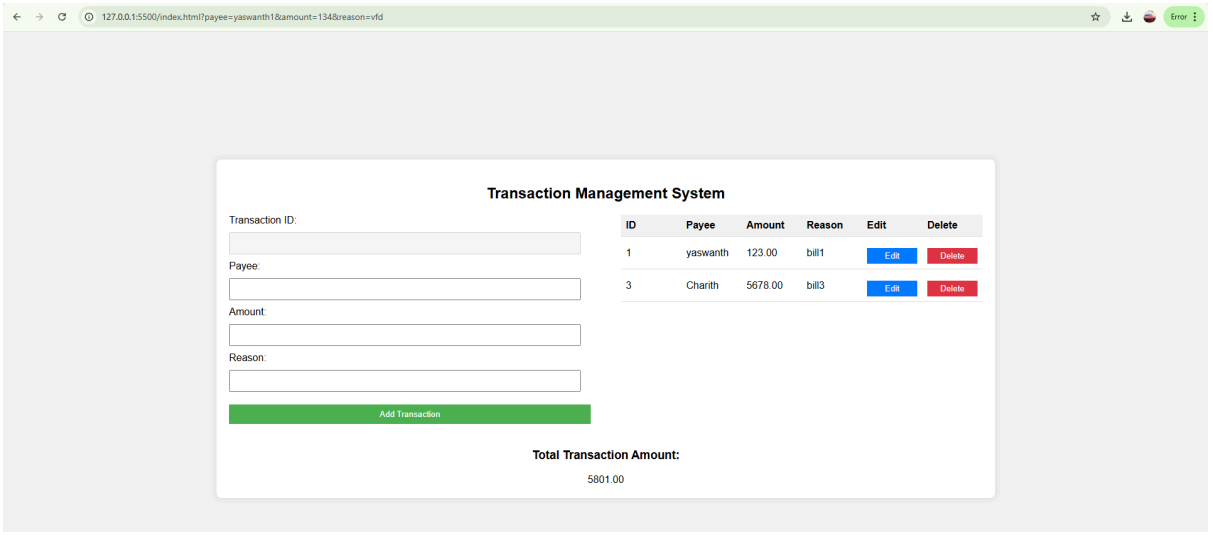
```
    }
}

function editTransaction(id) {
    const transaction = transactions.find(t => t.id === id);
    if (transaction) {
        document.getElementById('payee').value = transaction.payee;
        document.getElementById('amount').value = transaction.amount;
        document.getElementById('reason').value = transaction.reason;

        isEditing = true;
        editingTransactionId = id;
        document.querySelector('button[type="submit"]').textContent =
'Edit Transaction';
    }
}

async function deleteTransaction(id) {

    const index = transactions.findIndex(t => t.id === id);

    if (index !== -1) {
        // task6: Complete the below statement to remove a transaction
from the 'transactions' array by the index value
        transactions.splice(index, 1);

        updateTransactionTable();
        await updateTotalTransactionAmount();
    }
}
// task7: Convert below function 'updateTotalTransactionAmount' to use
async/await.
/*
 a. Convert the 'updateTotalTransactionAmount' function to an async
function.
 b. Update all occurrences of 'updateTotalTransactionAmount' across the
script to use await and ensure their calling functions are marked as
async.
*/
async function updateTotalTransactionAmount() {
    let totalAmount = 0;
    transactions.forEach(transaction => {
        totalAmount += transaction.amount;
```

```
    });

    document.getElementById('totalTransactionAmount').textContent =
totalAmount.toFixed(2);
}


function updateTransactionTable() {
    const tbody = document.querySelector('#transactionTable tbody');
    tbody.innerHTML = '';

    if (transactions.length === 0) {
        const noTransactionMessage = document.createElement('tr');
        noTransactionMessage.innerHTML = `<td colspan="6">No
Transactions found.</td>`;
        tbody.appendChild(noTransactionMessage);
    } else {
        transactions.forEach(transaction => {
            const row = document.createElement('tr');
            row.innerHTML = `
                <td>${transaction.id}</td>
                <td>${transaction.payee}</td>
                <td>${transaction.amount.toFixed(2)}</td>
                <td>${transaction.reason}</td>
                <td><button class="edit-button"
onclick="editTransaction(${transaction.id})">Edit</button></td>
                <td><button class="delete-button"
onclick="deleteTransaction(${transaction.id})">Delete</button></td>
            `;
            tbody.appendChild(row);
        });
    }
}


function clearForm() {
    document.getElementById('payee').value = '';
    document.getElementById('amount').value = '';
    document.getElementById('reason').value = '';
}
```

JavaScript Output



After deleting the transaction



After transaction edit