**Question 2: Bank Management System**

**Index.html**

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Bank Management System</title>

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

</head>

<body>

  <div id="login-section">

    <h2>Login to Your Bank Account</h2>

    <input type="text" id="account-name" placeholder="Enter your name">

    <button id="login-btn">Login</button>

    <button id="create-account-btn">Create New Account</button>

  </div>

  <div id="account-section" style="display: none;">

    <h2>Welcome, <span id="account-holder-name"></span></h2>

    <p>Your current balance is: $<span id="balance">0</span></p>

    <input type="number" id="deposit-amount" placeholder="Amount to deposit">

    <button id="deposit-btn">Deposit</button>

    <input type="number" id="withdraw-amount" placeholder="Amount to withdraw">

    <button id="withdraw-btn">Withdraw</button>

    <button id="logout-btn">Logout</button>

  </div>

  <script type="module" src="main.js"></script>

</body>

</html>

**Styles.css:**

**Code:**

body {

    font-family: 'Georgia', serif;

    background-color: #f4f4f4;

    margin: 0;

    padding: 0;

    text-align: center;

}

h2 {

    margin-top: 30px;

    color: #333;

    font-weight: bold;

}

#login-section, #account-section {

    margin-top: 50px;

    background-color: #ffffff;

    border-radius: 12px;

    box-shadow: 0 6px 12px rgba(0, 0, 0, 0.15);

    padding: 30px;

    max-width: 450px;

    margin-left: auto;

    margin-right: auto;

}

input {

    margin: 15px 0;

    padding: 12px;

    font-size: 18px;

    border: 1px solid #ccc;

    border-radius: 6px;

    width: calc(100% - 30px);

}

input:focus {

    border-color: #007BFF;

    outline: none;

}

button {

    padding: 12px 24px;

    font-size: 18px;

    cursor: pointer;

    background-color: #007BFF;

    color: white;

    border: none;

    border-radius: 6px;

    transition: background-color 0.3s ease, transform 0.2s ease;

}

button:hover {

    background-color: #0056b3;

    transform: translateY(-2px);

}

button:active {

    transform: translateY(1px);

}

p {

    color: #555;

}

#account-holder-name {

    font-weight: bold;

}

**Main.js**

**Code:**

import BankAccount from './bankAccount.js';

let currentAccount = null;

let accounts = JSON.parse(localStorage.getItem('accounts')) || {};

Object.keys(accounts).forEach(accountName => {

  const account = accounts[accountName];

  accounts[accountName] = new BankAccount(account.accountHolder, account.balance);

});

const loginSection = document.getElementById('login-section');

const accountSection = document.getElementById('account-section');

const accountNameInput = document.getElementById('account-name');

const depositInput = document.getElementById('deposit-amount');

const withdrawInput = document.getElementById('withdraw-amount');

const balanceDisplay = document.getElementById('balance');

const accountHolderNameDisplay = document.getElementById('account-holder-name');

document.getElementById('login-btn').addEventListener('click', () => {

  const accountName = accountNameInput.value.trim();

  if (accounts[accountName]) {

    currentAccount = accounts[accountName];

    loginSection.style.display = 'none';

    accountSection.style.display = 'block';

    accountHolderNameDisplay.innerText = currentAccount.accountHolder;

    balanceDisplay.innerText = currentAccount.balance;

  } else {

    alert('Account not found. Please create an account first.');

  }

});

document.getElementById('create-account-btn').addEventListener('click', () => {

  const accountName = accountNameInput.value.trim();

  if (accountName && !accounts[accountName]) {

    const newAccount = new BankAccount(accountName);

    accounts[accountName] = newAccount;

    localStorage.setItem('accounts', JSON.stringify(accounts));

    alert('Account created successfully! You can now log in.');

  } else {

    alert('Invalid name or account already exists.');

  }

});

document.getElementById('deposit-btn').addEventListener('click', () => {

  const amount = parseFloat(depositInput.value);

  if (isNaN(amount) || amount <= 0) {

    alert('Please enter a valid deposit amount.');

  } else {

    const message = currentAccount.deposit(amount);

    balanceDisplay.innerText = currentAccount.balance;

    alert(message);

    localStorage.setItem('accounts', JSON.stringify(accounts));

  }

});

document.getElementById('withdraw-btn').addEventListener('click', () => {

  const amount = parseFloat(withdrawInput.value);

  if (isNaN(amount) || amount <= 0) {

    alert('Please enter a valid withdrawal amount.');

  } else {

    const message = currentAccount.withdraw(amount);

    balanceDisplay.innerText = currentAccount.balance;

    alert(message);

    localStorage.setItem('accounts', JSON.stringify(accounts));

  }

});

document.getElementById('logout-btn').addEventListener('click', () => {

  loginSection.style.display = 'block';

  accountSection.style.display = 'none';

  accountNameInput.value = '';

});

**bankAccount.js:**

**Code:**

export default class BankAccount {

    constructor(accountHolder, initialBalance = 0) {

      this.accountHolder = accountHolder;

      this.balance = initialBalance;

    }

    deposit(amount) {

      if (amount > 0) {

        this.balance += amount;

        return `Deposited $${amount}. New balance: $${this.balance}`;

      }

      return 'Deposit amount must be positive.';

    }

    withdraw(amount) {

      if (amount > 0 && this.balance >= amount) {

        this.balance -= amount;

        return `Withdrew $${amount}. New balance: $${this.balance}`;

      } else if (amount <= 0) {

        return 'Withdrawal amount must be positive.';

      } else {

        return 'Insufficient balance for this withdrawal.';

      }

    }

    checkBalance() {

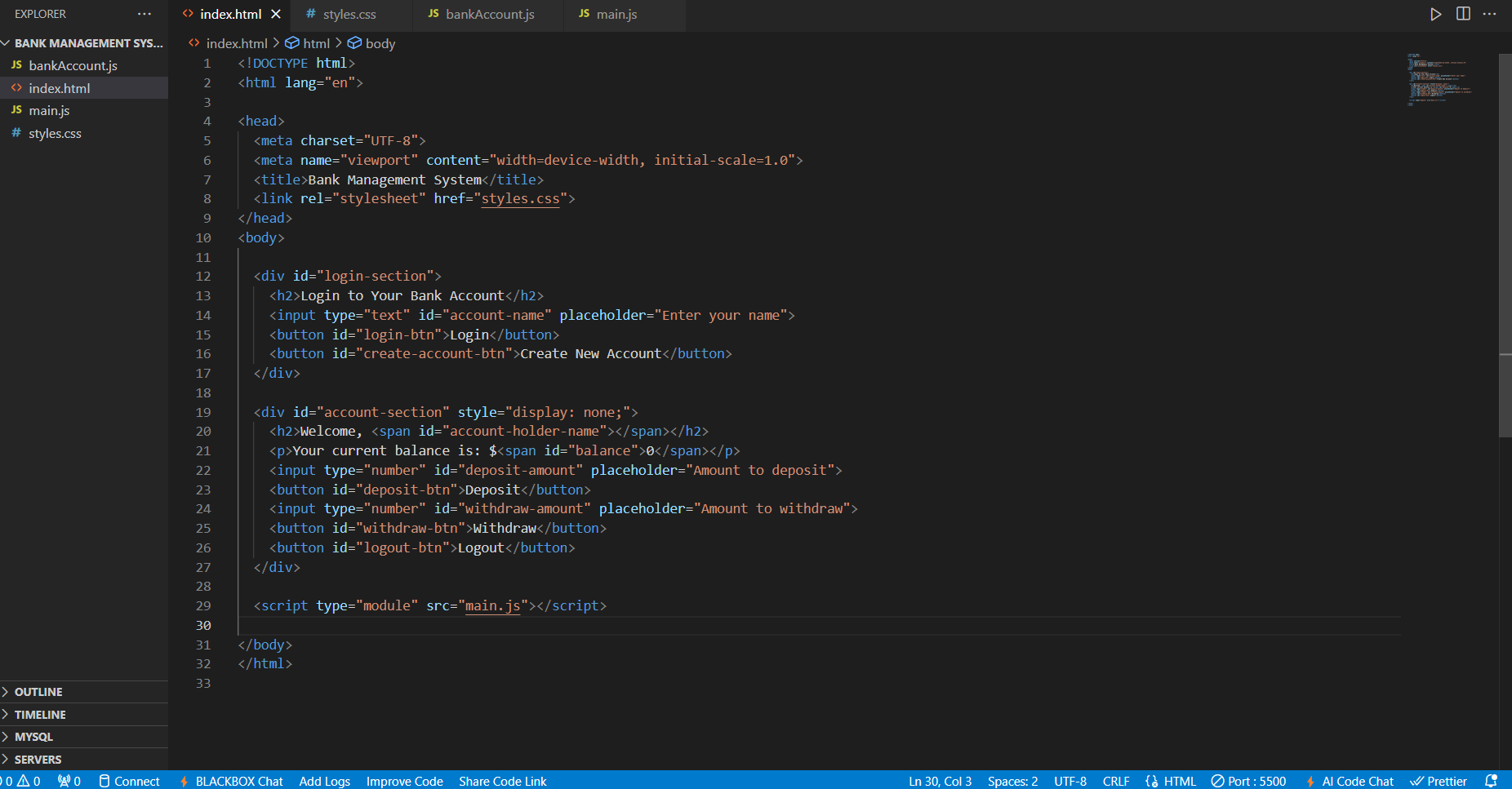
      return `Current balance: $${this.balance}`;

    }

  }

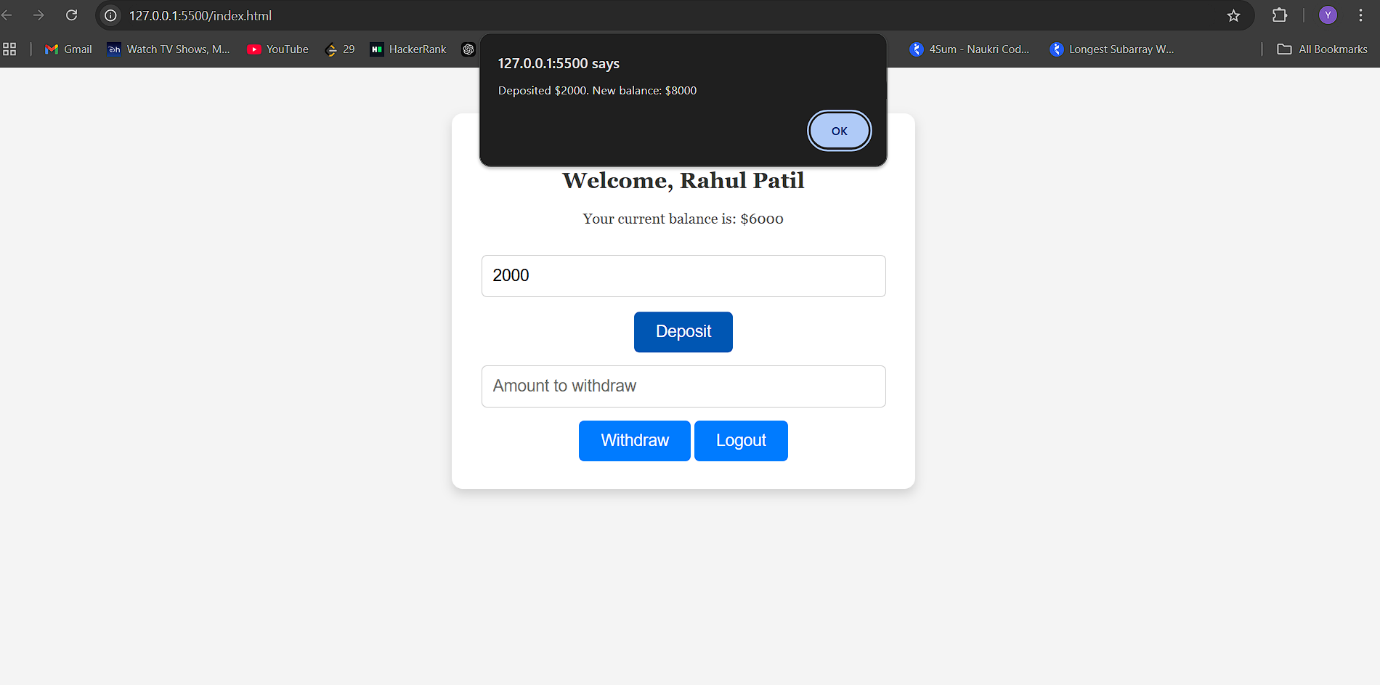
**OUTPUTS:**

**ES6 module Structure:**

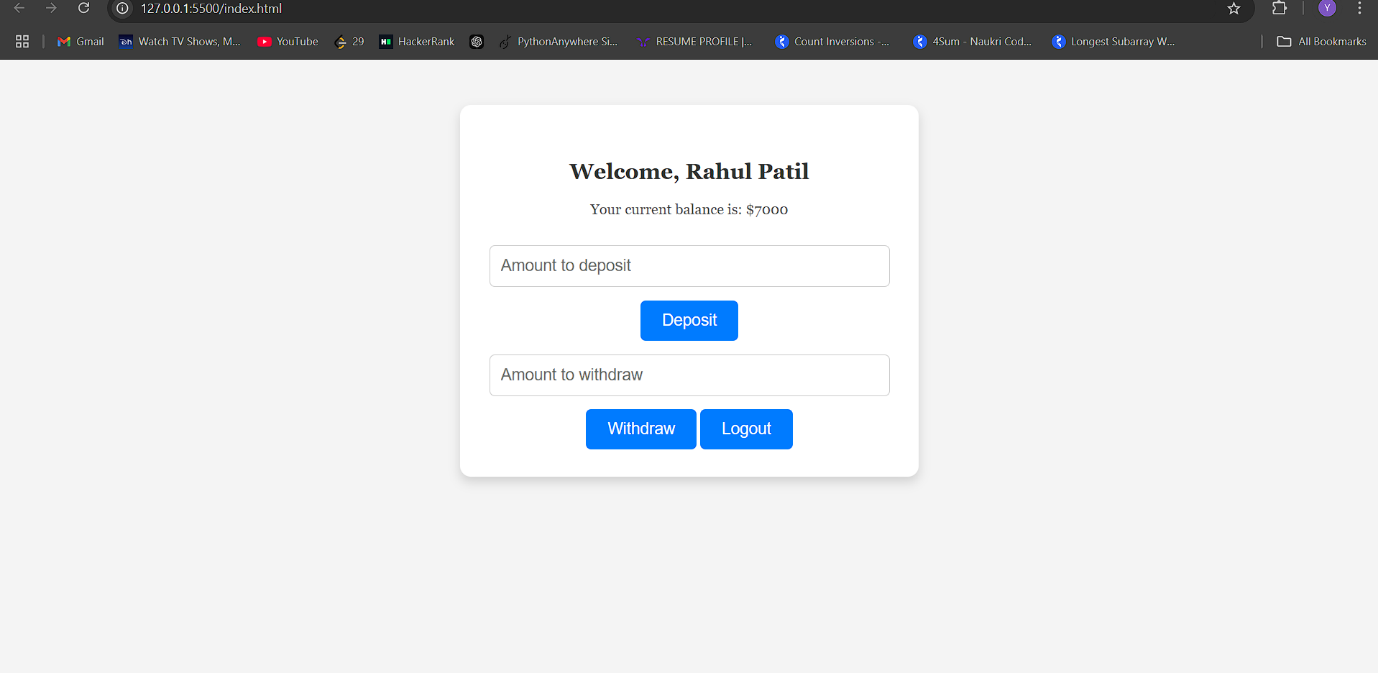


**Functional Interactions:**

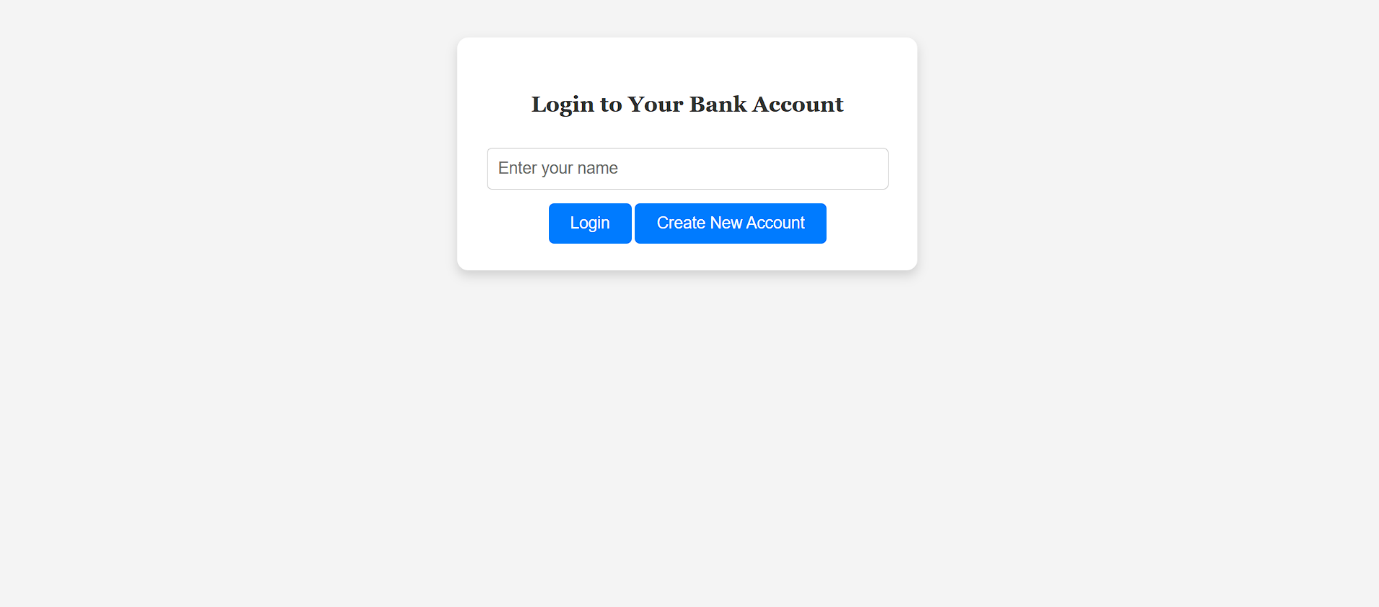
**Account creation:**



**Current Balance:**



**Deposit:**



**Withdrawal:**

