**Team Budget Planner**

**Code:**

**Index.html:**

<!DOCTYPE html>

<html>

  <head>

    <meta charset="utf-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <title>Team Budget</title>

    <meta name="description" content="" />

    <meta name="viewport" content="width=device-width, initial-scale=1" />

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

  </head>

  <body>

    <h2><b>TEAM BUDGET PLANNER</b></h2>

    <div class="container">

      <h4>Your balance</h4>

      <h1 id="balance">$0.0</h1>

      <div class="inc-exp-container">

        <div>

          <h4>income</h4>

          <p id="money-plus" class="money plus">+$0.00</p>

        </div>

        <div>

          <h4>expense</h4>

          <p id="money-minus" class="money minus">-$0.00</p>

        </div>

      </div>

      <h3>History</h3>

      <ul id="list" class="list">

      </ul>

      <h3>add new transation</h3>

      <form action="" id="form">

        <div class="form-control">

          <label for="text">text</label>

          <input type="text" id="text" placeholder="please enter the text..." />

        </div>

        <div class="form-control">

          <label for="amount">amount</label

          >

          <input

            type="number"

            id="amount"

            placeholder="please enter the amount..."

          />

        </div>

        <button class="btn">Add transation</button>

      </form>

    </div>

    <script src="script.js" async defer></script>

  </body>

</html>

**Style.css**

@import url("https://fonts.googleapis.com/css2?family=Viaoda+Libre&display=swap");

:root {

  --bg: #acbabc;

  --text: #0d0d0d;

  --line: #ebd4d4;

  --last: #5d5959;

  --box-shadow: 0 1px 5px rgba(10, 10, 10, 0.2),

    0 1px 5px rgba(255, 255, 255, 0.3);

}

\* {

  box-sizing: border-box;

}

body {

  font-family: "Viaoda Libre", cursive;

  background-color: var(--bg);

  color: var(--text);

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  margin: 0;

  min-height: 100vh;

}

.container {

  margin: 30px auto;

  width: 350px;

}

h1 {

  letter-spacing: 1px;

  margin: 0;

}

h3 {

  border-bottom: 1px solid var(--line);

  padding: 10px;

  margin: 40px 0 10px;

}

h4 {

  margin: 0;

  text-transform: uppercase;

  color: var(--last);

}

p {

  color: var(--bg);

  font-weight: bold;

}

.inc-exp-container {

  background-color: var(--line);

  padding: 20px;

  box-shadow: var(--box-shadow);

  display: flex;

  justify-content: space-between;

  margin: 20px 0;

}

.inc-exp-container > div {

  flex: 1;

  text-align: center;

}

.inc-exp-container > div:first-of-type {

  border-right: 1px solid var(--last);

}

.money {

  font-size: 20px;

  letter-spacing: 1px;

  margin: 5px 0;

}

.money.plus {

  color: forestgreen;

}

.money.minus {

  color: crimson;

}

label {

  display: inline-block;

  margin: 10px 0;

  font-size: large;

  font-weight: bold;

  text-decoration: underline;

}

input[type="text"],

input[type="number"] {

  border: 1px solid var(--last);

  border-radius: 5px;

  display: block;

  font-size: 16px;

  padding: 10px;

  width: 100%;

}

.btn {

  cursor: pointer;

  background-color: var(--line);

  box-shadow: var(--box-shadow);

  color: var(--bg);

  display: block;

  font-size: 16px;

  font-weight: bold;

  margin: 10px 0 30px;

  padding: 10px;

  width: 100%;

}

.btn:focus,

.delete-btn:focus {

  outline: none;

}

.list {

  list-style-type: none;

  padding: 0;

  margin-bottom: 40px;

}

.list li {

  background-color: var(--text);

  color: var(--bg);

  display: flex;

  justify-content: space-between;

  position: relative;

  padding: 10px;

  margin: 10px 0;

}

.list li.plus {

  border-right: 5px solid forestgreen;

}

.list li.minus {

  border-right: 5px solid crimson;

}

.delete-btn {

  cursor: pointer;

  background-color: var(--line);

  border: 0;

  font-size: 20px;

  line-height: 20px;

  padding: 2px 5px;

  position: absolute;

  top: 50%;

  left: 0;

  transform: translate(-100%, -50%);

  opacity: 0;

  transition: opacity 0.3s ease-in;

}

.list li:hover .delete-btn {

  opacity: 1;

}

**Script.js**

const money\_plus = document.getElementById("money-plus");

const list = document.getElementById("list");

const form = document.getElementById("form");

const text = document.getElementById("text");

const amount = document.getElementById("amount");

const money\_minus = document.getElementById("money-minus");

const localStorageTransations = JSON.parse(localStorage.getItem("transations"));

let transations =

  localStorage.getItem("transations") !== null ? localStorageTransations : [];

//add transation

function addTransation(e) {

  e.preventDefault();

  if (text.value.trim() === "" || amount.value.trim() === "") {

    text.placeholder = "please add a text";

    text.style.backgroundColor = "#ccc";

    amount.placeholder = "please add amount";

    amount.style.backgroundColor = "#ccc";

  } else {

    const transation = {

      id: genenrateID(),

      text: text.value,

      amount: +amount.value,

    };

    transations.push(transation);

    addTransationDOM(transation);

    updateValues();

    updateLocalStorage();

    text.value = "";

    amount.value = "";

  }

}

//generate id

function genenrateID() {

  return Math.floor(Math.random() \* 100000000);

}

//add transations to dom list

function addTransationDOM(transation) {

  //get sign

  const sign = transation.amount < 0 ? "-" : "+";

  const item = document.createElement("li");

  //add class based on value

  item.classList.add(transation.amount < 0 ? "minus" : "plus");

  item.innerHTML = `${transation.text} <span>${sign}${Math.abs(

    transation.amount

  )}</span> <button class="delete-btn" onclick="removeTransation(${

    transation.id

  })">x</button>`;

  list.appendChild(item);

}

//update the balance

function updateValues() {

  const amounts = transations.map((transation) => transation.amount);

  const total = amounts.reduce((acc, item) => (acc += item), 0).toFixed(2);

  const income = amounts

    .filter((item) => item > 0)

    .reduce((acc, item) => (acc += item), 0)

    .toFixed(2);

  const expense = (

    amounts.filter((item) => item < 0).reduce((acc, item) => (acc += item), 0) \*

    -1

  ).toFixed(2);

  balance.innerText = `$${total}`;

  money\_plus.innerText = `$${income}`;

  money\_minus.innerText = `$${expense}`;

}

//remove

function removeTransation(id) {

  transations = transations.filter((transation) => transation.id !== id);

  updateLocalStorage();

  init();

}

//updatelocal storage

function updateLocalStorage() {

  localStorage.setItem("transations", JSON.stringify(transations));

}

//init

function init() {

  list.innerHTML = "";

  transations.forEach(addTransationDOM);

  updateValues();

}

init();

form.addEventListener("submit", addTransation);