

Expense Tracker -A Web-Based Tool for Efficient Financial Management

Submitted By

Student Name	Student ID
NAKIB AL HASAN SABBIR	221-15-4760

MINI LAB PROJECT REPORT

This Report Presented in Partial Fulfillment of the course **CSE324:**
Operating System Lab in the Computer Science and Engineering
Department



DAFFODIL INTERNATIONAL UNIVERSITY

Dhaka, Bangladesh

December 10, 2024

DECLARATION

We hereby declare that this lab project has been done by us under the supervision of **Mr. Deawan Rakin Ahamed Remal, Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere as lab projects.

Submitted To:


10.12.2024

Mr. Deawan Rakin Ahamed Remal

Lecturer

Department of Computer Science and Engineering

Daffodil International University

Submitted by

NAKIB AL HASAN SABBIR

NAKIB AL HASAN SABBIR

ID:221-15-4760

Dept. of CSE, DIU

COURSE & PROGRAM OUTCOME

The following course have course outcomes as following:

Table 1: Course Outcome Statements

CO's	Statements
CO1	Make use of the Linux commands and shell programming.
CO2	Apply different algorithms of Operating Systems like CPU scheduling algorithm, page replacement algorithms, deadlock avoidance, and detection algorithm with analyzing the performance of them.
CO3	Able to design and develop a course project that can have a positive impact on the environment or society.

Table 2: Mapping of CO, PO, Blooms, KP and CEP

CO	PO	Blooms	KP	CEP
CO1	PO1	C3	KP1, KP2	EP1
CO2	PO2	C3, C4	KP3, KP4	EP2, EP3
CO3	PO3, PO9	C4, A4, P4	KP5	EP5

The mapping justification of this table is provided in section **4.3.1**, **4.3.2** and **4.3.3**.

Table of Contents

Declaration	i
Course & Program Outcome	ii
1 Introduction	1
1.1 Introduction.....	1
1.2 Motivation	1
1.3 Objectives	1
1.4 Feasibility Study.....	2
1.5 Gap Analysis.....	2
1.6 Project Outcome	2
2 Proposed Methodology/Architecture	4
2.1 Requirement Analysis & Design Specification	4
2.1.1 Overview	4
2.1.2 Proposed Methodology/ System Design	4
2.1.3 UI Design	5
2.2 Overall Project Plan	7
3 Implementation and Results	8
3.1 Implementation	8
3.2 Performance Analysis	12
3.3 Results and Discussion	13
4 Engineering Standards and Mapping	14
4.1 Impact on Society, Environment and Sustainability	14
4.1.1 Impact on Life	14
4.1.2 Impact on Society & Environment	14
4.1.3 Ethical Aspects.....	14
4.1.4 Sustainability Plan	14
4.2 Project Management and Team Work	15
4.3 Complex Engineering Problem.....	15
4.3.1 Mapping of Program Outcome.....	16
4.3.2 Complex Problem Solving	16
4.3.3 Engineering Activities.....	17

5 Conclusion	18
5.1 Summary.....	18
5.2 Limitation	18
5.3 Future Work	19
References	20

Chapter 1

Introduction

At present, the above chapter narrates about the project. It proceeds to state with a brief introduction of the background and problem statement under which the Expense Tracker is working. Motivation, objective, feasibility and gap analysis, as well as expected outcome, are among topics discussed here.

1.1 Introduction

Expense Tracker is a web-based application designed to simplify the management of personal finances for Every Users. The project addresses the general need for a reliable, secure, and easy-to-use tool for revenue and expense management [1]. It is built using HTML, CSS and JavaScript and stores data locally. With the help of the program, users can easily register, login and manage their financial activities. One of the features is real-time calculation of total revenue, expenses and remaining balance; If expenses exceed income, an alarm is shown. Users can view a summary of their financial information as well as add, modify and remove transactions; All this is safely stored in the local storage of the browser [2]. By enabling many users to safely handle their data on their own, the system ensures data isolation and privacy. This project provides an accessible money management solution without requiring server-based infrastructure, demonstrating how modern online technologies may be utilized for personal reasons (S. Chandini1, 2019).

1.2 Motivation

The ability to manage one's money effectively is an important skill to acquire; yet most people have no idea of their earnings and expenses. The existing financial products are either too complicated, require access to the internet, or even expose users' private information to third-party services [1]. The main driver of this research is the demand for more effective offline solutions that take into account user privacy and usability. Developing an expense tracker will solve this problem and enhance frontend web development efficiency. Develop practical skills working with HTML, CSS and JavaScript to create responsive and visually appealing online apps [2].

It also demonstrates how to manage data in local storage using modern browser features. Since the project solves a very real problem, it not only improves the life of the consumer but also that of the developer. This shows how people can better manage their money with just this simple yet useful approach.

1.3 Objectives

Expense Tracker aims to accomplish the following objectives:

1. Build a secure online money management app to track income and expenses.
2. Allow user registration and login while protecting the privacy and security of your data.
3. Make it possible to add, edit and remove financial transactions.
4. It is necessary to generate and present real-time summaries of earnings, expenses, and outstanding balances.
5. Customers should be aware when their spending exceed their income so they may make wise financial decisions.

6. allows for multiple users while protecting the privacy and security of each user's data.

Use local storage to efficiently store and retrieve data offline without relying on server-side infrastructure.

This project addresses real-world money management issues and improves web construction skills, demonstrating how contemporary technology may provide useful and approachable answers to common problems.

1.4 Feasibility Study

Numerous financial management apps are available, including YNAB, Mint, and Google Sheets. Even though some apps have many features, they often require complicated setup, a live internet connection or subscription. Privacy issues also arise when clients have to provide sensitive financial information to third-party servers. The Expense Tracker project is a lightweight alternative where all data resides in the user's browser and requires no external servers to operate. In such a way, application design is eased while data protection and accessibility in offline mode are granted. Such comparative studies and applications document the increasing popularity of privacy-sensitive financial solutions [2]. Our solution uses an easy user interface and local storage features to ensure seamless data management for multiple users.

Expense Tracker is a representative of how modern online technology can meet personal financial needs without the complexity or risk of a server-based system.

1.5 Gap Analysis

1. Limited Data Persistence

- Currently, all data is being stored in local storage which can be cleared by a browser or lost when switching between devices. This further restricts long-term data persistence or even multi-device synchronization for the project.

2. No Data Backup or Recovery

- There is no way a user can back up and recover their data. Users can lose all their financial records if local storage is empty or corrupted.

3. Scalability constraints

- Because of the space restrictions, local storage may fail to handle numerous transactions efficiently. This can be a problem with customers who have long financial histories.

4. Lack of Value Addition in Performance

- The project does not provide such advanced features as segment-wise analysis, charts, and graphs that enhance the presentation of financial data.

5. Insufficiently secure encryption

- Although the application stores user data locally, this data is not encrypted and can be accessed by unauthorized users or malicious scripts on the same device.

6. Absence of mobile optimization

- For customers who prefer smartphones or tablets for tracking spending, the present interface might not be entirely adapted for mobile devices, limiting accessibility.

7. Insufficient User Input

- It would have been ideal if this project included more means of giving users feedback in case there is an error message, confirmation after the deletion or update of any transaction.

8. Lack of Multi-Currency Support

- This application supports only one type of currency. This will definitely limit users who need to deal with multiple types of currency.

9. Static Warning System

- The warning system currently only warns in cases when expenses outweigh income but does not provide further suggestions or automated insights to enable better financial management.

10. Dependency on Manual Input

- Transactions have to be added all manually, and there is no provision for importing data from external sources or the generation of automated recurring transactions.

1.6 Project Outcome

Expense Tracker is a thorough approach to managing personal finances. Users can securely register and sign in to manage their own financial transactions with assurance of data confidentiality and privacy. The project includes tools to add, change and remove transactions with a summary of revenue and expenses and a real-time balance calculation. In situations where consumers' income surpasses their expenses, it promotes responsible financial management. By ensuring offline access through local storage, the program solves the problem of server-side installation. The privacy of each user's data is protected, and it is easy to support many users. This project demonstrates how to leverage modern web technologies to tackle real-world problems and develop safe, practical apps. It offers a starting point for upcoming improvements, including implementing sophisticated analytics or switching to a database-driven system.

Ultimately, the application offers consumers a simple yet useful money management tool

Chapter 2

Proposed Methodology/Architecture

This chapter reports on the system architecture and methodology for developing the Expense Tracker application. It deals with requirement analysis, design specifications, user interface design, and overall project plan, thus giving a coherent structure to the implementation process.

2.1 Requirement Analysis & Design Specification

2.1.1 Overview

Expense Tracker is a web-based program that manages data locally and is powered by HTML, CSS and JavaScript. The system makes it easy and secure to track income and expenses of numerous people. It satisfies the need for a simple, portable money management program that ensures usability, privacy and offline access.

Required prerequisites include:

1. Authentication and user registration (including a password and email).
2. Data management provides add, modify and delete options for revenue and expense transactions.
3. Calculation of income, expenses and remaining balance in real time.
4. A warning system for overspending.
5. Support for multiple users and individual, isolated data storage.

2.1.2 Proposed Methodology/ System Design

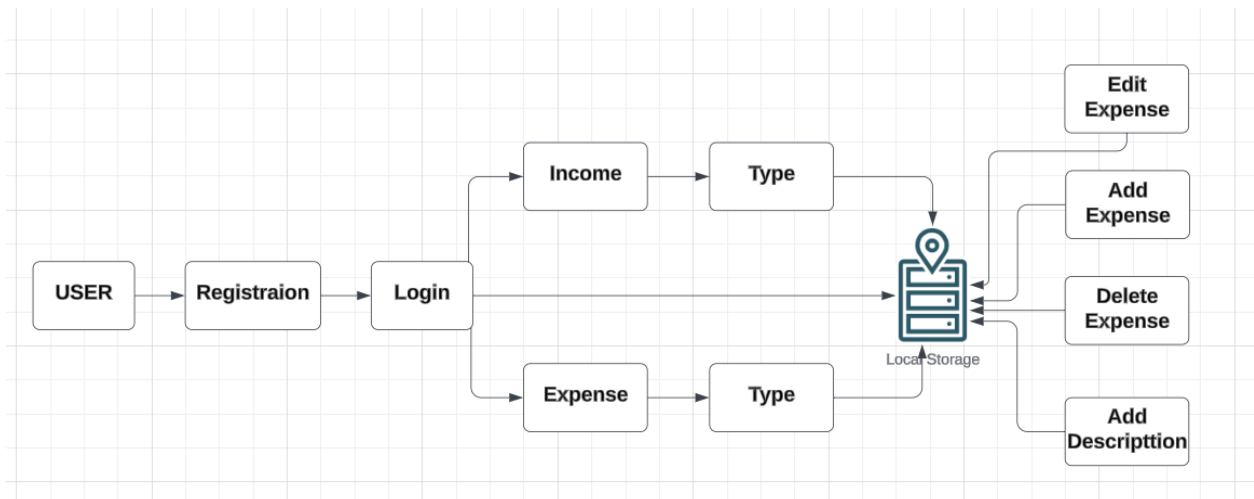
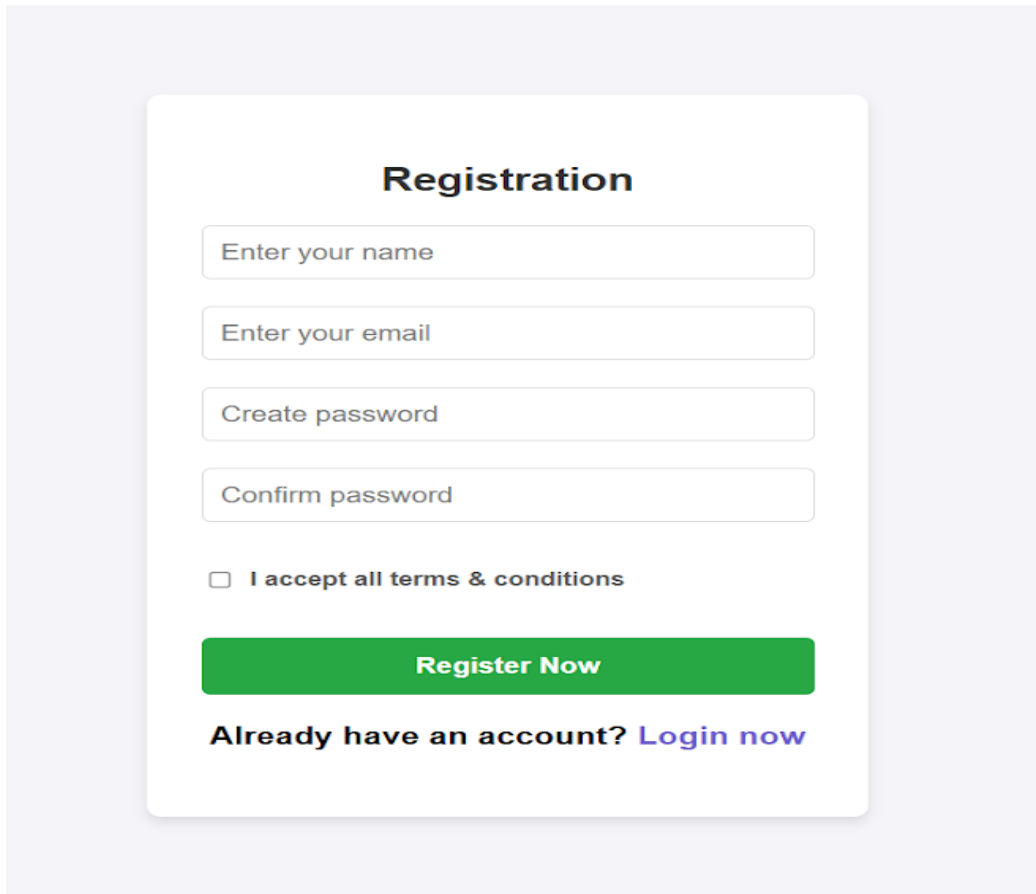


Figure 2.1: This is an Expense Diagram Design

2.1.3 UI Design

The UI design elements listed below prioritize simplicity and usability:

Registration and Login Page: Sign up and login forms require your name, email address and password.



The registration form is titled "Registration" in bold black text. It features four input fields with placeholder text: "Enter your name", "Enter your email", "Create password", and "Confirm password". Below the fields is a checkbox labeled "I accept all terms & conditions". A prominent green button labeled "Register Now" is positioned below the checkbox. At the bottom, the text "Already have an account? Login now" is displayed, with "Login now" in blue.

Registration

Enter your name

Enter your email

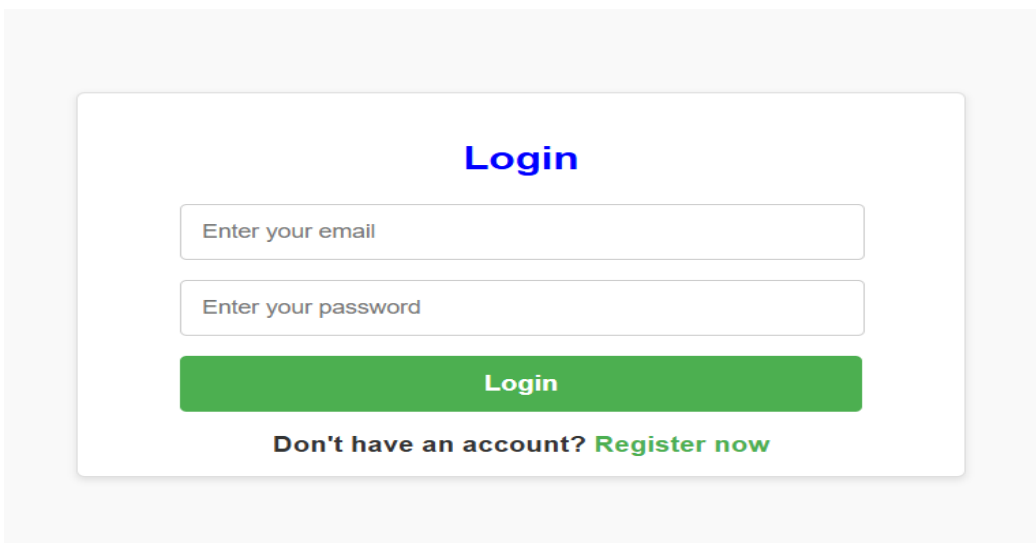
Create password

Confirm password

☐ I accept all terms & conditions

Register Now

Already have an account? [Login now](#)



The login form is titled "Login" in bold blue text. It features two input fields with placeholder text: "Enter your email" and "Enter your password". A prominent green button labeled "Login" is positioned below the fields. At the bottom, the text "Don't have an account? Register now" is displayed, with "Register now" in green.

Login

Enter your email

Enter your password

Login

Don't have an account? [Register now](#)

Dashboard: Balance, total revenue, total cost, and transaction summary are displayed on the dashboard.

The dashboard is titled "Expense Tracker" and includes a "Logout" button in the top right corner. The main summary box shows a "Balance: 3163.00". Below this, it displays "Total Income: 8500.00" and "Total Expenses: 5337.00". The "Transaction Summary" table lists three transactions: "Previous Due" (Expense, 837, 2024-12-05), "Pocket Money" (Income, 8500, 2024-12-02), and "Earphone" (Expense, 4500, 2024-12-05). Each transaction has "Edit" and "Delete" buttons. To the right is the "Add Transaction" form with fields for "Entry type" (a dropdown), "Name", "Date" (mm/dd/yyyy), and "Amount", followed by an "Add Transaction" button.

S.no.	Name	Amount	Date	Type	Update	Delete
1	Previous Due	837	2024-12-05	Expense	Edit	Delete
2	Pocket Money	8500	2024-12-02	Income	Edit	Delete
3	Earphone	4500	2024-12-05	Expense	Edit	Delete

- **Transaction form:** Name, date, amount, add/update/delete button and fields for transaction type (income/expense).
- **List of Transactions:** Shows each transaction with SL number, date, type, name and amount in tabular format.
- **Warning Message:** A popup warning is displayed when expenses are greater than income.
- **Logout Button:** A clear button that takes users to the login page after logging out.

The dashboard shows the same layout as the first screenshot, but with updated data. The "Balance" is now "-9637.00". The "Total Income" remains "8500.00", but the "Total Expenses" have increased to "18137.00". The "Transaction Summary" table now includes a fourth transaction: "Versity Fees" (Expense, 12800, 2024-12-08). A red warning message at the bottom right states "Warning: Expenses exceed income!".

S.no.	Name	Amount	Date	Type	Update	Delete
1	Previous Due	837	2024-12-05	Expense	Edit	Delete
2	Pocket Money	8500	2024-12-02	Income	Edit	Delete
3	Earphone	4500	2024-12-05	Expense	Edit	Delete
4	Versity Fees	12800	2024-12-08	Expense	Edit	Delete

2.2 Overall Project Plan

The project was developed in the following phases to ensure a structured approach:

- **Requirements gathering:** Identifies what the system needs to have. Usability, privacy, and data management have been focused on.
- **Design phase:** Wireframes of the system architecture and user interface were created to depict how the system would look and work.
- **Implementation:** HTML, CSS, and JavaScript were used for the development of the front end. Data management has been done using local storage.
- **Testing and Debugging:** Each feature was reviewed for appropriate data handling, a responsible user interface, and flawless execution.
- **Deployment:** The application was completed for use in browsers, ensuring offline functionality and user privacy. This will enable logically creating a secure and efficient expenditure tracking program in a very positive way.

Chapter 3

Implementation and Results

A detailed discussion regarding the implementation, performance analysis and results of the task management application is presented in this chapter.

3.1 Implementation

The Expense Tracker was developed in HTML, CSS, and JavaScript, while the data were managed through local storage. This project was also deployed on Netlify, so it would be much easier to access using a web browser.

Some key details of implementation include:

1. User Registration:

- The user needs to enter their name, email address, and password.
- JSON format is used for saving data in local storage for user information.

```
<> register.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Register</title>
7    <link rel="stylesheet" href="css/register.css">
8  </head>
9  <body>
10   <div class="box_container">
11     <h2 class="registration_style">Registration</h2>
12     <form id="registerForm">
13       <div class="input-box">
14         <input class="userinput" type="text" id="registerName" placeholder="Enter your name" required>
15       </div>
16       <div class="input-box">
17         <input class="userinput" type="email" id="registerEmail" placeholder="Enter your email" required>
18       </div>
19       <div class="input-box">
20         <input class="userinput" type="password" id="registerPassword" placeholder="Create password" required>
21       </div>
22       <div class="input-box">
23         <input class="userinput" type="password" id="registerConfirmPassword" placeholder="Confirm password" required>
24       </div>
25       <div class="policy">
26         <input type="checkbox" id="registerTerms" required>
27         <h3>I accept all terms & conditions</h3>
28       </div>
29       <div class="input-box button">
30         <input type="submit" value="Register Now">
31       </div>
32       <div class="text">
33         <h3>Already have an account? <a href="login.html">Login now</a></h3>
34       </div>
35     </form>
36   </div>
37   <script src="js/validation.js"></script>
38 </body>
39 </html>
```

```

JS validation.js > ...
// Registration validation check
document.getElementById("registerForm").addEventListener("submit", function (e) {
    e.preventDefault();

    const name = document.getElementById("registerName").value;
    const email = document.getElementById("registerEmail").value;
    const password = document.getElementById("registerPassword").value;
    const confirmPassword = document.getElementById("registerConfirmPassword").value;
    const terms = document.getElementById("registerTerms").checked;

    if (password !== confirmPassword) {
        alert("Passwords do not match!");
        return;
    }

    if (!terms) {
        alert("You must accept the terms and conditions!");
        return;
    }

    // Save data to local storage
    const users = JSON.parse(localStorage.getItem("users")) || [];
    const existingUser = users.find((user) => user.email === email);

    if (existingUser) {
        alert("User already exists with this email!");
        return;
    }

    users.push({ name, email, password });
    localStorage.setItem("users", JSON.stringify(users));
    alert("Registration successful! Redirecting to login.");
    // Here it redirect to login page
    window.location.href = "login.html";
});

```

2. Login

- Enter your registered Email address and password.
- The Credentials are checked against the ones stored in the local storage. It redirects the users to a dashboard after checking.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <link rel="stylesheet" href="css/login.css">
</head>
<body>
    <div class="box_container">
        <h2 class="login_design">Login</h2>
        <form id="loginForm">
            <div class="userinput">
                <input type="email" id="loginEmail" placeholder="Enter your email" required>
            </div>
            <div class="userinput">
                <input type="password" id="loginPassword" placeholder="Enter your password" required>
            </div>
            <div class="userinput button">
                <input type="submit" value="Login">
            </div>
            <div class="text">
                <h3>Don't have an account? <a href="register.html">Register now</a></h3>
            </div>
        </form>
    </div>
    <script src="js/validation.js"></script>
</body>
</html>

```

```

// Login validation check
document.getElementById("loginForm")?.addEventListener("submit", function (e) {
    e.preventDefault();

    const email = document.getElementById("loginEmail").value;
    const password = document.getElementById("loginPassword").value;

    const users = JSON.parse(localStorage.getItem("users")) || [];
    const user = users.find((user) => user.email === email && user.password === password);

    if (!user) {
        alert("Invalid email or password!");
        return;
    }

    // Save session data (logged-in user)
    localStorage.setItem("loggedInUser", JSON.stringify(user));
    alert("Login successful! Redirecting to the Expense Tracker.");
    // Redirect to index.html
    window.location.href = "index.html";
});

// Logout Logic
document.getElementById("logoutButton")?.addEventListener("click", function () {
    localStorage.removeItem("loggedInUser");
    alert("Logged out successfully!");
    // Redirect to login page
    window.location.href = "login.html";
});

// Redirect to login if not logged in
if (!localStorage.getItem("loggedInUser") && window.location.pathname.includes("index.html")) {
    alert("Please log in first!");
    // Redirect the login
    window.location.href = "login.html";
}

```

3. Dashboard:

- It showcases dynamically total income, total expenses and remaining balance
- Contains options: Add transaction, Transaction summaries, Logout.

```

<body>
  <header>
    <div class="navbar">
      <button id="logoutButton" class="logout-btn" onclick="logout()">Logout</button>
    </div>
  </header>
  <div>
    <h3 style="color: chocolate; font-size: 1.3rem;">Expense Tracker</h3>
  </div>
  <div class="summary">
    <div>
      <h1>Balance:
      <span id="updateBalance">0</span>
    </div>
    <br />
    <div class="total">
      <div>
        Total Income:
        <div>
          <h2 id="updateIncome">0</h2>
        </div>
      </div>
      <hr class="verticle" />
      <div>
        Total Expenses:
        <div>
          <h2 id="updateExpense">0</h2>
        </div>
      </div>
    </div>
  </div>
  <div class="root">
    <div id="items">
      <h2>Transaction Summary</h2>
      <!-- Add wrapper for scrolling bar -->
      <div id="table-container">
        <table id="table">
          <tr class="titles">
            <th>S.no.</th>
            <th>Name</th>
            <th>Amount</th>
            <th>Date</th>
            <th>Type</th>
            <th>Update</th>
            <th>Delete</th>
          </tr>
        </table>
      </div>
    </div>
    <hr class="vertical" />
    <div id="new" class="add-transaction-container">
      <h2>Add Transaction</h2>
      <div class="inputs">
        <div class="inputitem">
          <label for="itemType">Entry type:</label>
          <select id="itemType">
            <option value="" disabled selected>Select type</option>
            <option value="0">Expense</option>
            <option value="1">Income</option>
          </select>

```

```

let transactions = []; // Store transactions globally

// Save transactions for the logged-in user
function saveToLocalStorage() {
  const loggedInUser = JSON.parse(localStorage.getItem("loggedInUser"));
  if (loggedInUser) {
    // Save transactions specific to the logged-in user
    localStorage.setItem(`transactions_${loggedInUser.email}`, JSON.stringify(transactions));
  }
}

// Load transactions from localStorage when page loads
window.onload = function () {
  const loggedInUser = JSON.parse(localStorage.getItem("loggedInUser"));
  if (loggedInUser) {
    // Load transactions for the logged-in user
    const storedTransactions = localStorage.getItem(`transactions_${loggedInUser.email}`);
    if (storedTransactions) {
      transactions = JSON.parse(storedTransactions);
    }
  } else {
    // If no user is logged in, redirect to login page
    window.location.href = "login.html";
  }

  renderTable();
};

// Add a new transaction
function addItem() {
  const type = document.getElementById("itemType").value;
  const name = document.getElementById("name").value;
  const amount = document.getElementById("amount").value;
  const date = document.getElementById("date").value;

  if (!type || !name || !amount || !date) {
    alert("Please fill out all fields.");
    return;
  }

  transactions.push({ type, name, amount, date });
  saveToLocalStorage(); // Save after adding
  renderTable();
  clearForm();
}

// Delete a transaction
function deleteItem(index) {
  transactions.splice(index, 1);
  saveToLocalStorage(); // Save after deleting
  renderTable();
}

// Edit a transaction
function editItem(index) {
  const transaction = transactions[index];
  document.getElementById("itemType").value = transaction.type;
  document.getElementById("name").value = transaction.name;
  document.getElementById("amount").value = transaction.amount;
  document.getElementById("date").value = transaction.date;

  transactions.splice(index, 1);
  saveToLocalStorage(); // Save after editing
}

```


4. Transaction management: Users are able to add transactions on typing.

1. Type: Income or Expense
2. Name: What is being bought
3. Date
4. Amount
 - Data to be stored on local storage with visual tabulation.
 - User updates or deletes transactions to avoid mistakes.

5. Alert System: If the total expenses exceed income, a warning message is shown until expenses are reduced.

6. Logout Functionality: It clears session data and redirects the user to the login page.

7. Deployment on Netlify: The project was deployed on Netlify, ensuring that users can access the application from any browser.

3.2 Performance Analysis

The following are usability, functionality, and performance reviews of the Expense Tracker:

Strengths:

1. Multiple User Support:

Local storage is utilized to store data for each user independently, thereby eliminating the occurrence of version conflicts.

2. Offline Functionality:

Since the application uses local storage, it is reliable in offline mode, where one would not have access to the internet.

- **Ease of Use:**

The application interface is very simple and easy to understand; thus, it is accessible to all users.

- **Time Calculations:**

The dashboard dynamically updates the income, expenses, and remaining balance.

- **Error Handling:**

The features like updating and deletion of the transactions enhance its usability.

Limitations:

- **Data Persistence:**

The local storage is browser-specific; thus, clearing browser cache wipes out all data.

- **Security Issues:**

The data in local storage is not encrypted, thus vulnerable to malicious scripts.

- **Scalability Issues:**

The capacity of local storage is limited, thus may not allow the usability of this application for users with extensive data.

- **No Advanced Features**

Lacks data visualization tools like charts or graphs that might be useful.

3.3 Results and Discussion

Following is the set of results derived through the Expense Tracker:

- **Conflict less multi user management:**

In registering, logging in and managing their financial data there are no conflicts or overwriting.

- **Effective management of transaction:**

The application's ability to add and edit or delete the transaction makes the tracking of the financial data easy and flexible in nature.

- **Enhanced knowledge of finances:**

Users can monitor the expenditures, hence avoid overpaying because the system provides real time summarized views and warning messages.

- **Offline Accessibility:**

A potential ability of this application working via offline storage.

Chapter 4

Engineering Standards and Mapping

This chapter is supposed to outline the engineering aspects of the Expense Tracker project. It is expected to take into consideration societal, environmental, and ethical impacts of the project, a sustainability plan for the project, an assessment and recommendation of project management strategies, mapping to complex engineering problem-solving criteria and justification to specific program outcomes.

4.1 Impact on Society, Environment and Sustainability

4.1.1 Impact on Life

The Expense Tracker will improve financial literacy and decision-making through effective tracking of one's income and expenses. This will foster personal and family financial stability by encouraging better budgeting habits.

4.1.2 Impact on Society & Environment

- **Social impact:** The project teaches money management and financial responsibility, an essential aspect of the economic health of any society.
- **Environmental Impact:** The program saves the environment by managing money digitally, thus reducing the use of paper for maintaining physical records.

4.1.3 Ethical Aspects

The initiative has provided a proper ethical usage for the data by segregating the data of the user and has allowed access through credentials only. However, data security can be updated for better security, such as encryption, to modern standards.

4.1.4 Sustainability Plan

The following methods are used by the project to guarantee sustainability:

- The project's use of browser-based local storage lessens its reliance on the server.
- Physical resource consumption will decrease if people are encouraged to use digital tools. By combining cloud storage and encryption techniques, future scaling can be accomplished while maintaining long-term relevance and use.

4.2 Project Management and Team Work

Budget Considerations:

1. Cost Analysis

This Expense Tracker project would have had very negligible expenditures, being developed using free tools HTML, CSS, JavaScript, and Local Storage, deployed on Netlify. Development time was probably the only cost, estimated between \$500-\$600 for taking 50-60 hours at \$10 per hour. In an alternative budget, premium hosting, say with Firebase, at about \$20-\$30 per month and design tools like Figma at \$15-\$20 per month could allow better scalability and user experience.

2. Revenue Model

In this case, the freemium model may be an option, where basic features are provided at no cost, and users should pay for advanced tools such as analytics, cloud backup, or more. This will probably generate income by means of subscription plan or one-time fee for licensing. This might also include ad-related income to cover costs for the users. All put together, a balanced sustainability and value for the users could be obtained from these models.

4.3 Complex Engineering Problem

4.3.1 Mapping of Program Outcome

This project would illustrate how the outcomes of such an academic program could solve real-world problems. For example:

(PO's):

Table 4.1: Justification of Program Outcomes

PO's	Justification
PO1	Engineering knowledge
PO2	Problem analysis
PO3	Design /Development of solutions
PO4	Investigation
PO5	Modern tools usages
PO6	The engineer and the society
PO7	Environment and sustainability
PO8	Ethics
PO9	Individual work and teamwork
PO10	Communication
PO11	Project management and finance
PO12	Life-long learning

4.3.2 Complex Problem Solving

PO1: Engineering Knowledge

The Expense Tracker project demonstrates profound engineering knowledge in how such basic technologies as HTML, CSS, and JavaScript can be used to construct an interactive and dynamic web application. Using this tool, responsiveness of the user interface and continuity of functionality are conveyed. Light data management techniques are put to practical use as evidenced by the use of local storage, so the application works sans backend database. The knowledge applied in the development of this tool demonstrates an understanding of principles of software architecture and development principles.

PO2: Problem Analysis

The project systematically analyzes the general problem of people's financial mismanagement. The solution is usable and effective in addressing problems such as secure user authentication, transaction tracking, and dynamic calculation of income, expenses, and remaining balance. The analysis was hence necessary in finding out what the user wanted and incorporation of features such as overspending notifications and categorized tracking of transactions.

PO3: DESIGN/DEVELOPMENT OF SOLUTIONS

It has been designed in such a way that it considers the users at each and every step by facilitating access and ensuring efficiency. Important features include security of data through user authentication, management of transactions for cash inflow and outflow, updating of balance on real-time basis, scalable architecture for further enhancements, etc. Every single feature has been developed for robustness, ranging from storage to small UI elements. These solutions collectively meet the needs of the users and satisfy the project objectives.

PO4: Research

In the development process, much investigation and deep analysis were done to eliminate problems and hone the solution. The application would have gone through comprehensive testing against targeted problems with a view to establishing an error-free security regime against miscalculating or manipulating of data inefficiently. When finalized, this project will guarantee its dependability and smoothness for all its lot of customers through debugging and optimization at each stage through the application cycle.

PO5: Modern Tool Usage

The use of modern tools and technologies was of great help to this project. Netlify was used for smooth deployment, so the user could directly access the application through web browsers without any extra installations. JavaScript was used to ensure real-time updates and dynamic functionality, while local storage was used for the efficient storing of user data. These tools, in their own way, enhanced the performance, accessibility, and scalability of the application by adhering to modern paradigms of web development.

PO6: The Engineer and the Society

The project solves a societal need by offering an accessible and secure platform for personal financial management. It will enable users to track their financial activities and develop better spending habits. It will contribute to the financial well-being of its users by offering a free, user-friendly solution that encourages responsible money management practices.

PO7: Environment and Sustainability

This is an environmental minimum of impact since it involves no physical infrastructure or major resources from the server. The project uses local storage to avoid unnecessary data transfers, trying to reduce energy use in any possible way. It also provides for very light architecture for long-term sustainability, therefore being environmentally friendly in respect to financial management.

PO8: Ethics

The project observes ethical standards, such as maintaining data privacy and security. User authentication prevents sensitive financial information from unauthorized access. Fixed goals have been set in design/architecture to ensure that users are confident and transparent about their personal information, according to various sets of ethical guidelines on software development. This depicts the care for the users' information and integrity in that respect.

PO9: Individual Work and Teamwork

The individual effort and capability related to designing, coding, and testing a comprehensive solution are emphasized in the development of this project. The project shows the ability of independent working with all the flexibility to include team contribution if required in future also. The project makes a right platform for collaboration by adopting modular and adaptable design.

PO10: COMMUNICATION

The project emphasizes effective communication through proper documentation and structuring of its objectives, features, and outcomes. This project logically organizes the information in such a way that any stakeholder, whether a user or an evaluator, will be able to perceive the purpose and functionalities of the project with ease. This improves user experience and makes any future development discussion clear as crystal.

PO11: Project Management and Finance

Efficient project management made the development of the application within both time and resource constraints. Hosting via Netlify, costs are waived, which is very practical for budget management. The modularity of the development made tracking easier, while aligning to the various milestones of the project so as to assure a smooth and organized workflow.

PO12: Life-long Learning

The project was based on learning new skills of web development, such as the deployment of applications in Netlify, JavaScript to make the features dynamic, and optimization of local storage to store data. These experiences will be added to lifelong learning and show how much it is necessary to adapt to emerging technologies to create innovative solutions.

Table 4.2: Mapping with complex problem solving.

EP1 Dept of Knowledge	EP2 Range of Conflicting Requirement s	EP3 Depth of Analysis	EP4 Familiarity of Issues	EP5 Extent of Applicable Codes	EP6 Extent Of Stakeholder Involvement	EP7 Inter- dependence
Thus, demonstrates almost perfect command over the web technologies HTML, CSS, and JavaScript to create an extremely feature-rich interface and a friendly interaction experience with the user.	Presents a balance of simplicity versus scalable, maintainable structures while addressing user needs and the appropriate technical requirements.	To thoroughly visit the offline-first design on local storage for analyzing non-persistent data usability in-depth	These well-known problems are solving things such as handling user interaction and smooth offline availability through tried-and-true techniques and technologies.	Make good and as well maintainable codes by adhering to the required coding standards and best practices in web development.	The course steers the iterative design and development processes, through tuning in its user feedback and usability testing, towards the realities of the expected world.	With regards to data management and what goes on the front end, vast, uninterrupted integration of different components guarantees interdependence and system-wide coherence.

4.3.3 Engineering Activities

Table 4.3: Mapping with complex engineering activities.

EA1 Range of resources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for society and environment	EA5 Familiarity
Usage of intent and various web development tools such as libraries, frameworks and browser tools, in seeking cost-effective development.	To ensure very high levels of interaction with the users, the seamless design of the UI is coupled with very smooth transition and switching between offline and online modes.	Employs a pioneering offline-first mechanism utilizing browser local storage to meet the user's experience.	Instils responsible coding practices and accessibility so that customers are positively affected and engage digitally.	It takes familiar technologies and paradigms to ease development and speeded diagnosis, hence minimizing risks and delays.

Chapter 5

Conclusion

This chapter provides a comprehensive overview of the Expense Tracker project, its limitations, and its possible future improvements in order to make it useful and user-friendly.

5.1 Summary

Expense Tracker will be an **HTML, CSS, and JavaScript-based** lightweight web application utilizing local storage for data handling. Through a straightforward interface, it will make it simple for the user to track revenue, expenses, and the remaining balance in real time. As a result, users can independently and securely create an account and log in to view or change their financial data without jeopardizing the confidentiality and integrity of numerous users. Features include adding, updating, and deleting transactions; included is the added functionality of warnings if expenses surpass income.

The project's primary goals of increasing financial literacy and simplifying personal budgeting have been accomplished. It is guaranteed that Netlify deployments will reach a large audience in a way that is both useable and accessible. All things considered; the project is a helpful tool for managing finances while also setting the stage for future development.

5.2 Limitation

While the Expense Tracker achieves most of the primary objectives set out for it, the following limitations were noted:

1. **Data Persistence:** This will be bound to the browser and device it is on, and if the browser cache is cleared, it is deleted.
2. **Security Issues:** Information stored within the local storage is not encrypted and thus may pose a potential risk if the browser gets compromised.
3. **Scalability:** Local storage has size limits, and it cannot store an immense amount of data; neither can it handle thousands and thousands of transactions.
4. **Advanced Features:**
 - The application does not provide data visualization features like charts or graphs that would give a better insight.
 - No backup or cross-device synchronization options are included.
5. **Mobile Optimization:**
 - The application does not provide data visualization features like charts or graphs that would give a better insight.
 - No backup or cross-device synchronization options are included.

5.3 Future Work

It solves these issues in several ways: for example,

1. Cloud-Based Data Storage:

- Using cloud-based databases like Firebase or MongoDB enables cross-device synchronization and makes data durable.
- Data security, Sensitive user data, encrypted and kept locally or in the cloud, enhances security and privacy.

2. Scalable Architecture:

- Move to a highly scalable storage solution that will handle users with a very long history of transactions.

3. Advanced Analytics:

- Add charting, graphing, and trending to give users further insight into their spending habits.

4. Mobile-First Design:

- Optimize the application interface for mobile devices to ensure a seamless experience across a wide array of screen sizes.

5. Incorporate User Feedback:

Establish a feedback mechanism for users to provide suggestions for the continuous improvement of the system. These will further improve the application's usability, scalability, and effectiveness to keep it a useful tool in financial management.

References

- [1] Chandini, S., Poojitha, T., Ranjith, D., Akram, V.M., Vani, M.S. and Rajyalakshmi, V., 2019. Online Income and Expense Tracker. International Research Journal of Engineering and Technology (IRJET), 6(3), pp.2395-0056.
- [2] Bhatele, P., Mahajan, D., Mahajan, B., Mahajan, D., Mahajan, N. and Mahajan, P., 2023, May. TrackEZ Expense Tracker. In 2023 4th International Conference for Emerging Technology (INCET) (pp. 1-5). IEEE.
- [3] <https://www.scribd.com/document/407723189/Expense-Tracker>
- [4] Blog: [Features of Expense Tracker](#)

ORIGINALITY REPORT

10%

SIMILARITY INDEX

9%

INTERNET SOURCES

1%

PUBLICATIONS

8%

STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	4%
2	Submitted to University of Northumbria at Newcastle Student Paper	1%
3	www.coursehero.com Internet Source	1%
4	elearn.daffodilvarsity.edu.bd Internet Source	1%
5	Submitted to Kingston University Student Paper	1%
6	Submitted to Ajou University Graduate School Student Paper	1%
7	Submitted to Higher Education Commission Pakistan Student Paper	<1%
8	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1%
9	www.equinoxpub.com	

Internet Source

<1 %

10

www.ijrte.org

Internet Source

<1 %

11

www.propulsiontechjournal.com

Internet Source

<1 %

12

open.uct.ac.za

Internet Source

<1 %

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off