DAILY EXPENSE TRACKER

A Mini Project-I Report

Submitted to
RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA
BHOPAL (M.P)

MINI PROJECT REPORT
Submitted by

Ashish Kumar Choudhary [Roll No. 95] Ayush Kaushik [Roll No. 105]
Ark Kumar [Roll No. 86]

Under the supervision of
DR. NEERAJ TANTUBAY

Department of Computer Science Engineering

Lakshmi Narain College of Technology, Bhopal (M.P.)

Session
2023-24

INDEX

CHAPTER 1
PROBLEM DOMAIN DESCRIPTION

In our modern, fast-paced lives, effective management of personal finances is a paramount concern. The Expense Tracker Web Application serves as a solution, offering users a streamlined platform to monitor and categorize daily expenditures. By facilitating the input of expenses with details such as amount, date, and description, the application empowers users to gain insights into their spending patterns. Intuitive data visualization tools, including charts and graphs, provide a comprehensive overview of daily, weekly, and monthly expenses. Users can set and track budgets for various categories, receiving real-time updates and notifications to ensure financial goals are met. The system further incorporates search and filtering functionalities for efficient expense retrieval. With a responsive design, the web application ensures accessibility across diverse devices. Security measures, including encrypted data transmission and secure user authentication, safeguard sensitive financial information. Through features like export options and automatic backups, users can maintain accurate records of their financial transactions. The Expense Tracker Web Application aims to simplify financial management, foster informed decision-making, and encourage responsible spending habits for individuals striving towards financial stability.

About The Project

The Expense Tracker Web Application is a user-friendly platform designed to help individuals manage their daily expenses efficiently. Developed as a college mini project using HTML, CSS, and JavaScript, the application provides a secure user authentication system, allowing users to register accounts and log in securely. Once logged in, users can input their daily expenses, categorize them for better organization, and visualize spending patterns through charts and graphs. The application also supports budgeting by enabling users to set monthly limits for different expense categories, with real-time updates and notifications to ensure adherence to financial goals.

The intuitive design includes features such as search and filtering options, making it easy for users to locate specific expenses quickly. With a responsive layout, the web application ensures accessibility on various devices. Security measures, including encrypted data transmission and secure user authentication, protect sensitive financial information. Additionally, the application promotes user engagement through export options for expense data and automatic backups, providing a comprehensive solution for individuals seeking to cultivate better spending habits and make informed financial decisions. This project demonstrates a practical application of web development skills, combining functionality and usability to address real-world challenges in personal finance management.

CHAPTER 2
LITERATURE SURVEY

A literature survey on expense tracking and financial management reveals a growing interest in leveraging technology to address personal finance challenges. Several studies emphasize the importance of budgeting and tracking expenses for individuals to achieve financial goals and build financial resilience. Web-based applications, like the one developed in this project, play a crucial role in providing accessible tools for users to manage their finances effectively.

Research indicates that the integration of data visualization tools in expense tracking applications enhances user understanding of financial patterns. Visual representations, such as charts and graphs, contribute to improved financial literacy and decision-making. Additionally, studies highlight the significance of budgeting features in such applications, emphasizing their impact on user awareness and adherence to financial plans.

Security is a paramount concern in financial applications, as highlighted in various literature. The implementation of secure user authentication, data encryption, and protective measures against cyber threats is crucial to building user trust and safeguarding sensitive financial information.

Moreover, the literature recognizes the importance of user engagement features, such as notifications and reminders, to encourage consistent usage of expense tracking tools. Mobile responsiveness is another aspect highlighted, acknowledging the prevalence of smartphones and the need for applications to be accessible across various devices.

In conclusion, the literature survey underscores the relevance and significance of expense tracking applications in the context of personal financial management. This project aligns with the trends and recommendations identified in the literature, incorporating essential features to provide users with a comprehensive tool for effective expense tracking and financial planning.

CHAPTER 3
MAJOR OBJECTIVES & SCOPE OF PROJECT

3.1 Major Objectives:

3.1.1 Efficient Expense Tracking:

- Develop a user-friendly platform for individuals to input and track their daily expenses efficiently.

3.2.2 Data Visualization:

- Implement intuitive data visualization tools, such as charts and graphs, to offer users a clear overview of their spending patterns over different time frames.

3.3.3 Budget Management:

- Enable users to set monthly budgets for various expense categories, with real-time updates and notifications to help them stay within their financial limits.

3.4.4 User Authentication and Security:

- Ensure secure user authentication to protect personal financial data.
- Implement encryption measures to safeguard sensitive information and instill user confidence in the application's security.

3.5.5 Search and Filtering Functionality:

- Provide users with the ability to easily search and filter their expenses, enhancing the accessibility of specific financial information.

3.6.6 Responsive Design:

- Develop a responsive web application accessible across various devices, including desktops, tablets, and smartphones, to accommodate diverse user preferences.

3.2 Scope:

The scope of the project encompasses the development of a comprehensive Expense Tracker Web Application that caters to the diverse needs of individuals seeking effective personal financial management. The application will focus on providing a seamless user experience, combining ease of use with advanced features to encourage consistent and informed financial decision-making. The project's scope includes the integration of secure authentication, data visualization tools, budget management features, and responsive design, making it adaptable to users' preferences and accessible across different devices. Additionally, the scope involves implementing search and filtering functionalities for efficient data retrieval and user engagement features to enhance the application's usability. The project aims to create a valuable tool for individuals striving to cultivate better spending habits, adhere to financial goals, and achieve overall financial stability.

CHAPTER 4

PROBLEM ANALYSIS AND REQUIREMENT SPECIFICATION

4.1 Problem Analysis:

4.1.1 Financial Management Challenges:

- Many individuals face challenges in effectively managing their personal finances, leading to difficulties in budgeting and achieving financial goals.

4.1.2 Lack of Tracking Tools:

- There is a need for accessible and user-friendly tools to track daily expenses, as traditional methods may be cumbersome and prone to errors.

4.1.3 Data Overload:

- Users often struggle with managing and understanding large sets of financial data, hindering their ability to make informed decisions.

4.1.4 Security Concerns:

- In the era of increasing cyber threats, there is a growing concern regarding the security of personal financial information in online applications.

4.1.5 Limited Budget Awareness:

- Individuals may lack awareness of their monthly budget utilization, leading to overspending and financial strain.

4.2 Requirement Specification:

4.2.1 User Authentication:

- *Objective:* Implement a secure user authentication system.
- *Requirements:* User registration, login/logout functionality, password protection, and account recovery mechanisms.

4.2.2 Expense Entry and Categorization:

- *Objective:* Allow users to input daily expenses with relevant details.
- *Requirements:* Input form for expense details (amount, date, description), categorization options (e.g., groceries, transportation).

4.2.3 Data Visualization:

- *Objective:* Provide users with clear visualizations of their spending patterns.
- *Requirements:* Charts and graphs displaying daily, weekly, and monthly expenditure, with interactive features for enhanced user understanding.

4.2.4 Budget Management:

- *Objective:* Enable users to set and monitor monthly budgets.
- *Requirements:* Budget input functionality, real-time updates, notifications for budget limits.

4.2.5 Responsive Design:

- *Objective:* Ensure accessibility across various devices.
- *Requirements:* Responsive layout design, compatibility with desktops, tablets, and smartphones.

4.2.6 User Engagement Features:

- *Objective:* Encourage consistent usage and proactive financial management.
- *Requirements:* Notifications and reminders for regular expense tracking, achievement milestones.

CHAPTER 5
DETAILED DESIGN (MODELLING AND ERD/DFD)

5.1 Architecture and File Structure:

The project follows a simple web architecture with HTML, CSS, and JavaScript for the frontend. Here's a breakdown of the file structure:

5.1.1 Home Page:

- **index.html:** The main landing page displaying expense categories and providing links to login, signup, and data pages.
- **style.css:** Styles for the homepage.

5.1.2 Login Page:

- **login.html:** A simple login form.
- **login.css:** Styles for the login page.
- **login.js:** JavaScript for handling login functionality.

5.1.3 Signup Page:

- **signup.html:** Form for user registration.
- **signup.css:** Styles for the signup page.
- **signup.js:** JavaScript for handling user registration.

5.1.4 Data:

- **Bills, Entertainment, Grocery, Other Expenses Subfolders:**
  - **{category}.html:** HTML templates for displaying category-specific expense data.
  - **{category}.css:** Styles for the specific category page.
  - **{category}.js:** JavaScript for rendering expense data and creating a chart.

5.2 Data Flow:

5.2.1 User Registration:

- User registration data is collected using the signup form.
- Upon submission, the data is processed by **signup.js**.
- The user's data is stored in the **localStorage** with the email as the key.

5.2.2 User Login:

- User login details are entered through the login form.
- **login.js** checks if the entered credentials match the stored data in **localStorage**.
- If successful, the user is redirected to the home page with a user parameter in the URL.

5.2.3 Expense Tracking:

- The homepage allows users to input expenses for different categories.
- The expenses are stored in the user's data object in **localStorage**.

5.2.4 Category-specific Pages:

- Each category (Bills, Entertainment, Grocery, Other Expenses) has its HTML, CSS, and JavaScript files.
- The JavaScript retrieves and displays the user's expense data for the specific category.
- A chart is generated using Chart.js to visualize the expenses.

5.3 Entities Involved:

5.3.1 User:

- Attributes: Email, Password.

5.3.2 Expense Category:

- Attributes: Grocery, Entertainment, Bills, Other Expenses.
- Each category has an array of expenses.

5.3.3 Expense:

- Attributes: Amount, Date/Time.

5.4 ERD (Entity-Relationship Diagram):

5.4.1 User Entity:

- Attributes: Email (Primary Key), Password.

5.4.2 Expense Category Entity:

- Attributes: CategoryID (Primary Key), CategoryName.

5.4.3 Expense Entity:

- Attributes: ExpenseID (Primary Key), Amount, Date/Time, CategoryID (Foreign Key), UserID (Foreign Key).

5.5 DFD (Data Flow Diagram):

5.5.1 Level 0 (Context Diagram):

- **Processes:** User Registration, User Login, Expense Tracking, Display Category Data.
- **Data Stores:** Local Storage.
- **External Entities:** User.

5.5.2 Level 1 (Detailed DFD):

- **Processes:**
  - Register User, Login User.
  - Track Expense, Display Category Data.
- **Data Stores:**
  - **localStorage** for user data and expenses.
- **Data Flows:**
  - User registration/login data, Expense data.

CHAPTER 6
HARDWARE/SOFTWARE PLATFORM ENVIRONMENT

6.1 Visual studio

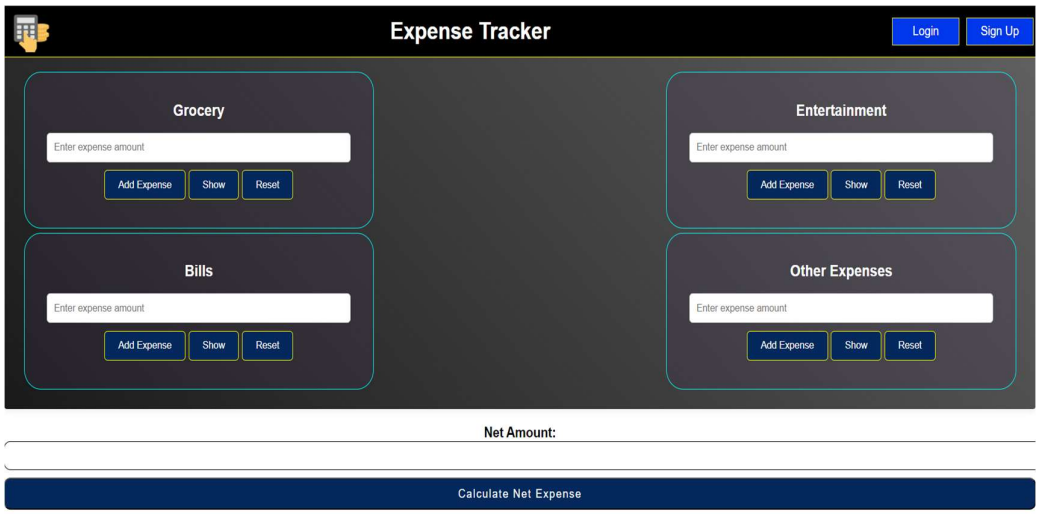
Visual Studio is an Integrated Development Environment (IDE) developed by Microsoft to develop GUI (Graphical User Interface), console, Web applications, web apps, mobile apps, cloud, and web services, etc. With the help of this IDE, you can create managed code as well as native code. It uses the various platforms of Microsoft software development software like Windows store, Microsoft Silver light, and Windows API, etc. It is not a language-specific IDE as you can use this to write code in C#, C++, VB (Visual Basic), Python, JavaScript, and many more languages. It is available for Windows as well as for macOS.

The first version of VS (Visual Stu”Io) ’as released in 1997, named as Visual Studio 97 having version number 5.0. The latest version of Visual Studio is 15.0 which was released on March 7, 2017. It is also termed as Visual Studio 2017. Java was supported in old versions of Visual Studio but in the latest version doesn’t provide any support for Java language.

CHAPTER 7
SNAPSHOT OF INPUT/OUTPUT

7.1 Expense Tracker Homepage

The Expense Tracker homepage is the user interface for the Expense Tracker Web Application, providing a clean and intuitive layout for users to manage their daily expenses.

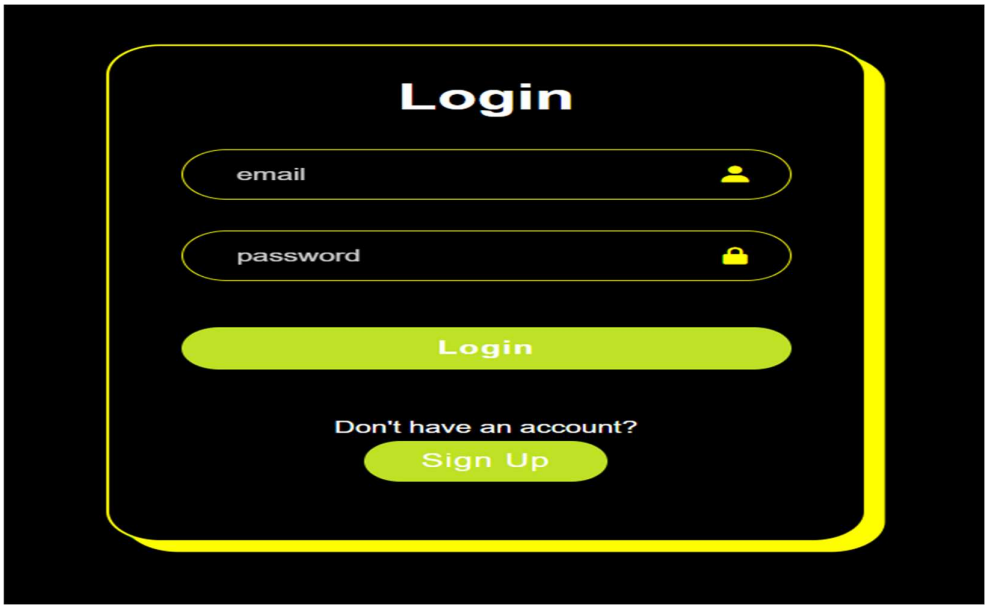


7.2 Expense Tracker Signup Page

The Expense Tracker Signup Page enables new users to create accounts and start managing their expenses.

7.3 Expense Tracker Login Page

The Expense Tracker Login Page provides users with a secure entry point to access their personalized expense tracking dashboard.



7.4 Expense Tracker History Page

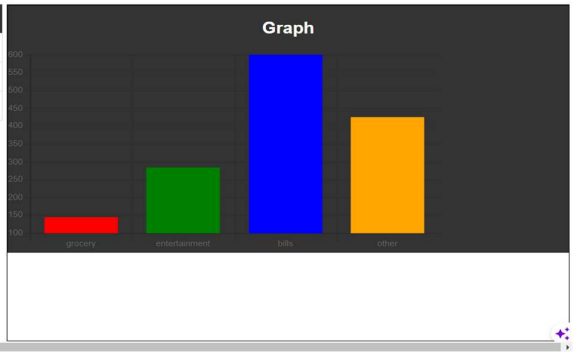
The Expense Tracker History Page is a crucial component of the web application, allowing users to review their past expenses in a comprehensive and visual manner.

Expense Data

Go Back

Entertainment

Date	Amount
2023-11-30T16:39:21.918Z	15
Thu, 30 Nov 2023 17:58:47 GMT	70
Thu, 30 Nov 2023 17:59:04 GMT	200

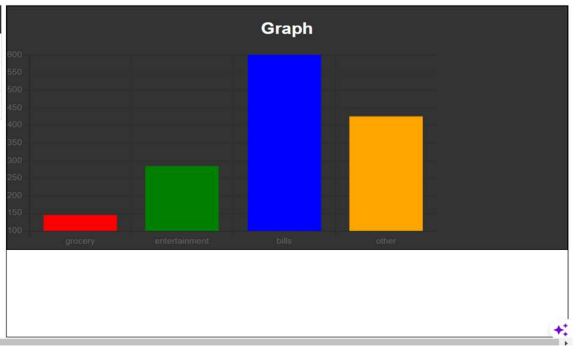


Expense Data

Go Back

Bills

Date	Amount
Thu, 30 Nov 2023 17:58:43 GMT	60
Thu, 30 Nov 2023 17:59:01 GMT	40
Thu, 30 Nov 2023 17:59:34 GMT	500

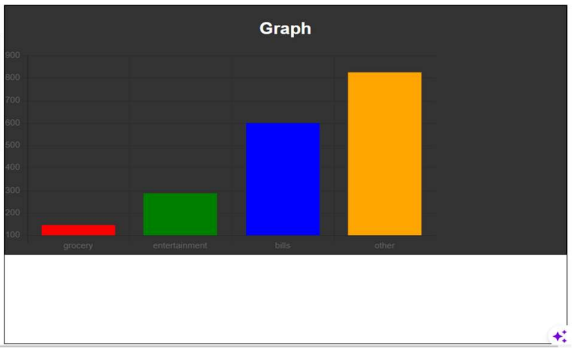


Expense Data

Go Back

Other Expenses

Date	Amount
2023-11-30T16:39:26.013Z	50
Thu, 30 Nov 2023 17:58:52 GMT	75
Thu, 30 Nov 2023 17:59:08 GMT	300
Thu, 30 Nov 2023 18:04:25 GMT	400



CHAPTER 8
CODING

8.1 Home Page

8.1.1 index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Expense Tracker</title>
    <link rel="stylesheet" type="text/css" href="style.css" />
  </head>
  <body>

    <div id="navbar">
      <img src="/images/expenseLogo.png" alt="expense_logo" />
      <h1>Expense Tracker</h1>
      <div id="btn" class="right-btn">
        <button><a href="/LoginPage/login.html" id="login"></a></button>
        <button><a href="/SignUpPage/signup.html" id="signup">Sign Up</a></button>
      </div>
    </div>

    <div class="container">
        <section class="expense-category">
            <h2>Grocery</h2>
            <input type="text" id="grocery" placeholder="Enter expense amount" />
            <button id="b1" onclick="addExpense('grocery') ">Add Expense</button>
            <button class="show" onclick="show('grocery')">Show</button>
            <button class="reset" onclick="reset('grocery')">Reset</button>
            <ul id="groceryList"><p id="para1"></p></ul>
        </section>

        <section class="expense-category">
            <h2>Entertainment</h2>
            <input  type="text"  id="entertainment"  placeholder="Enter expense amount" />
```

8.1.2 style.css

```css
body {
  font-family: Arial, sans-serif;
  background-image: url("background.jpg");
  background-size: cover;
  background-position: center;
  margin: 0;
  padding: 0;
  overflow-x: hidden;
}

#navbar {
  position: fixed;
  top: 0;
  left: 0;
  right: 0;
  z-index: 1000;
  display: flex;
  align-items: center; /* Align vertically centered */
  justify-content: space-between;
  width: 100%;
  height: 65px;
  background-color: #000;
  border: 1px solid yellow;
}

#navbar img {
  margin-top: 5px;
  margin-left: 10px;
  width: 60px;
  height: 50px;
}

```

8.1.3 script.js

```
const urlParams = new URLSearchParams(window.location.search);
const user = urlParams.get("user");
let data = JSON.parse(localStorage.getItem(user));

let isLoggedIn = false;
if (data != null) {
document.getElementById("login").innerHTML="Logout"
} else {
  document.getElementById("login").innerHTML="Login"
}

const expenses = {
  grocery: data ? data.expense.grocery : [],
  entertainment: data ? data.expense.entertainment : [],
  bills: data ? data.expense.bills : [],
  other: data ? data.expense.other : [],
};

function addExpense(categoryField) {
  const expenseAmountInput = document.getElementById(categoryField);
  const currentTime =  new Date().toUTCString();
    const expenseAmount = parseFloat(expenseAmountInput.value);
  if (!isNaN(expenseAmount)) {

    const prev = expenses[categoryField];
    const newObj = {
      time: currentTime,
      amount: expenseAmount
    }
    prev.push(newObj);
    const updatedData = {
      ...data,
```

8.2 Login Page

8.2.1 login.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link
      href="https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css"
      rel="stylesheet"
    />
    <link rel="stylesheet" href="login.css">
    <title>Login Page</title>
  </head>

  <body>
    <div class="wrapper">
      <form action="">
        <h1>Login</h1>
        <div class="input-box">
          <input  placeholder="email"   type="email"  id="email"   name="email" required />
          <i class="bx bxs-user"></i>
        </div>
        <div class="input-box">
          <input placeholder="password" type="password"  id="password"  name="password"  required />
          <i class="bx bxs-lock-alt"></i>
        </div>
        <div class="btn1">
          <button type="submit" onclick="move()">Login</button>
        </div>
        <div class="btn2">
          <p>Don't have an account?</p>
          <button id="b3" onclick="move1()">Sign Up</button>
        </div>
```

8.2.2 login.css

```css
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "poppins", sans-serif;
}
body {
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  background-color: black;
}
.wrapper {
  width: 420px;
  color: white;
  border-radius: 30px;
  padding: 30px 40px;
  border: 2px solid yellow;
  background-color: black;
  box-shadow: 10px 10px yellow,5px 5px green ;
}
.wrapper h1 {
  font-size: 36px;
  text-align: center;

}
.wrapper .input-box {
  position: relative;
  width: 100%;
  height: 50px;
  margin: 30px 0;
```

8.2.3 login.js

```
function move() {
  event.preventDefault();
  var y = document.getElementById("email").value;
  var t = document.getElementById("password").value;
  var data = localStorage.getItem(y);

  if (data == null) {
    alert("User Not found!!!!!");
  } else {
    parseData = JSON.parse(data);
    if (parseData.password == t) {
      window.location.assign(`/HomePage/index.html?user=${parseData.email}`);
      alert("Login Succesfull");
    } else {
      alert("Password is incorrect!!!");
    }
  }
}
```

8.3 Signup Page

8.3.1 signup.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="signup.css">
    <title>Signup Page</title>
</head>
<body>
        <div class="container">
            <div class="title">Sign up</div>
            <form action="your_server_endpoint_here" method="post">
                <div class="user-details">
                    <div class="input-box">
                        <span class="details">Full Name</span>
                        <input type="text" placeholder="Enter your name" required>
                    </div>
                    <div class="input-box" >
                        <span class="details">Username</span>
                        <input type="text" placeholder="Enter your Username"  required>
                    </div>
                    <div class="input-box">
                        <span class="details">Email</span>
                        <input type="text" placeholder="Enter your email" id="email" name="email" required>
                        <div class="error-box">
                            <div class="error">Invalid email address!</div>
                        </div>
                    </div>
                    <div class="input-box">
                        <span class="details">Phone Number</span>
                        <input type="tel" placeholder="Enter your number" required>
                        <div class="error-box">
```

8.3.2 signup.css

```css
*{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: 'poppins',sans-serif;
}
body{
    display: flex;
    height: 100vh;
    justify-content: center;
    align-items: center;
    background-color: black;
}
.container{
    max-width: 700px;
    width: 100%;
    background: #fff;
    padding: 25px 30px;
    border-radius: 30px;
    background-image: white;
    border: 1px solid yellow;
    background-color: black;
    box-shadow: 10px 10px yellow;

}
.container .title{
    font-size: 28px;
    font-weight: 600;
    position: relative;
    text-decoration: underline;
    color: white;
}
```

8.3.1 signup.js

```
function move2() {
  var q = document.getElementById("email").value;
  var w = document.getElementById("password").value;
  event.preventDefault();
  const expenses = {
    grocery: [],
    entertainment: [],
    bills: [],
    other: [],
  };
  const data = {
    email: q,
    password: w,
    expense: expenses,
  };
  localStorage.setItem(q, JSON.stringify(data));
  window.location.assign("/LoginPage/login.html");
}
```

8.4 Data Page

8.4.1 data.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Expense Data</title>
    <link rel="stylesheet" type="text/css" href="otherExpenses.css">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.5.0/Chart.min.js"></script>
</head>
<body>
    <div id="navbar">
        <h1>Expense Data</h1>
        <button id="backBtn" onclick="history.back()">Go Back</button>
    </div>
    <div id="heading">
        <h2>Other Expenses</h2>
    </div>
    <div class="container">

        <div class="left" id="categoryData">
            <table>
                <thead>
                    <tr>
                        <th>Date</th>
                        <th>Amount</th>
                    </tr>
                </thead>
                <tbody id="data-output">
                </tbody>
            </table>
        </div>
        <div class="right">
```

8.4.1 data.css

```css
body {
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
}
#navbar {
    background-color: #333;
    color: white;
    padding: 10px;
    text-align: center;
}
#heading{
    display: flex;
    align-items: center;
    justify-content: center;
    height: 50px;
    width: 100%;
    margin-top:2px;
    border: 1px solid yellow;
    background-color: black;
    color: white;
}
#backBtn {
    background-color: #4CAF50;
    color: white;
    border: none;
    padding: 10px 20px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
```

8.4.1 data.js

```
const urlParams = new URLSearchParams(window.location.search);
const user = urlParams.get("user");
const category = urlParams.get("category");

let data = JSON.parse(localStorage.getItem(user));

const categoryData = data.expense.other;


let placeholder = document.querySelector("#data-output");
let out = "";
for(let product of categoryData){
    out += `
      <tr>
        <td>${product.time}</td>
        <td>${product.amount}</td>
      </tr>
    `;
}

placeholder.innerHTML = out;

let groceryExpense=0;
let entertainmentExpense=0;
let billsExpense=0;
let otherExpenses=0;
data.expense.grocery.forEach(element => {
    groceryExpense += parseInt(element.amount);
});
data.expense.entertainment.forEach(element => {
    entertainmentExpense += parseInt(element.amount);
  });
```

CHAPTER 9
PROJECT LIMITATION AND FUTURE SCOPE

9.1 Project Limitations:

9.1.1 Offline Mode:

- The current version of the Expense Tracker operates in an online mode. Offline functionality, allowing users to input expenses without an internet connection, is not supported.

9.1.2 Limited Expense Categories:

- The application may have a predefined set of expense categories. Future versions could enhance flexibility by allowing users to create custom categories.

9.1.3 Basic Authentication:

- The current authentication system might lack advanced features like two-factor authentication or OAuth integration. Future improvements can enhance user account security.

9.1.4 Simplified Budgeting:

- The budget management feature is basic and may not cover advanced budgeting strategies. Future enhancements could include more sophisticated budgeting tools and financial planning features.

9.1.5 Device Compatibility:

- While the website is designed to be responsive, extensive testing on a wide range of devices and browsers may not have been conducted. Some users might experience inconsistencies in the user interface.

9.2 Future Scope:

9.2.1 Enhanced Analytics:

- Implement advanced analytics and data visualization tools to offer users more in-depth insights into their spending patterns, enabling better financial decision-making.

9.2.2 Machine Learning Integration:

- Explore the integration of machine learning algorithms to predict future expenses based on historical data, providing users with proactive financial insights.

9.2.3 Mobile Application Development:

- Extend the Expense Tracker to mobile platforms by developing dedicated iOS and Android applications, offering users a more convenient and accessible way to manage their expenses.

9.2.4 Expense Category Recommendations:

- Implement AI-driven features that analyze spending patterns and provide personalized recommendations for optimizing expense categories and budget allocations.

9.2.5 Integration with Financial APIs:

- Integrate the application with financial APIs to automatically fetch and categorize transactions from bank accounts, providing users with a more automated and seamless experience.

9.2.5 Advanced Security Features:

- Enhance security by implementing advanced authentication methods, encryption standards, and regular security audits to protect user data against evolving cyber threats.

9.2.6 Collaborative Budgeting:

- Introduce features that allow users to collaborate on budgeting with family or household members, facilitating a more comprehensive approach to managing shared expenses.

9.2.7 Expense Approval Workflows:

- Implement workflows that allow users to submit expenses for approval in a business or group setting, enhancing the application's utility in various contexts.

9.2.8 Expense Trend Notifications:

- Develop a notification system that alerts users to significant changes in spending patterns, helping them stay informed and make timely adjustments to their budgets.

9.2.9 Integration with Financial Education Resources:

- Include educational resources within the application to empower users with financial literacy and advice on better money management practices.

By addressing these future scopes, the Expense Tracker can evolve into a more sophisticated and comprehensive tool, providing users with advanced features and a richer financial management experience.

CHAPTER 10
REFRENCES

---

www.google.com
www.youtube.com
www.W3schools.com