**Web Development Intern Project Report**

* **Intern Name:** SHAIK NAJEER SULTHAN  
  **Internship Duration:** 06/06/2025 – 12/06/2025  
  **Position:** Web Development Intern  
  **Project Title:** Expense/Money Tracker Web Application

**1. Introduction**

During my internship as a Web Development Intern, I was assigned a project to develop a fully functional web application focused on real-world utility and enhanced user experience. The project required leveraging core web technologies — HTML, CSS, and JavaScript — to build an interactive, responsive, and user-friendly platform.

Depending on the project option chosen, the goal was either to help users track their finances effectively, provide an engaging landing page for a website or brand, or create a practical day planner to organize daily tasks.

**2. Project Objectives**

* Design and develop a clean, intuitive user interface (UI) using HTML5 and CSS3.
* Implement interactive features and dynamic content updates with JavaScript.
* Ensure the application is fully responsive and performs well on multiple devices and browsers.
* Incorporate client-side data storage using localStorage to persist user data.
* Enhance user experience (UX) with smooth navigation, feedback, and accessibility considerations.

**3. Project Descriptions and Features**

**Option 1: Expense/Money Tracker Web-App**

This application enables users to manage their personal finances by tracking incomes and expenses, categorizing transactions, and visualizing spending habits.

**Key Features:**

* **Transaction Management:** Add, edit, and delete income and expense entries with details like amount, category, date, and description.
* **Category Filters:** Group expenses by categories such as Food, Transport, Entertainment, Bills, etc.
* **Summary Dashboard:** Visual charts (pie chart, bar chart) displaying spending distribution and balance over time.
* **Budget Alerts:** Users can set monthly budgets and receive alerts if spending exceeds limits.
* **Responsive Design:** Mobile-first approach to enable ease of use on smartphones and tablets.
* **Data Persistence:** Use of browser localStorage to save user data locally without requiring backend setup.

**Option 2: Landing Page Development**

This project involves designing a landing page aimed at showcasing a product, service, or brand effectively to visitors, encouraging engagement or conversions.

**Key Features:**

* **Hero Section:** Catchy headline, background images or videos, and clear call-to-action (CTA) buttons.
* **Feature Highlights:** Sections detailing product benefits or service features with icons and images.
* **Testimonials:** User reviews or client feedback to build trust.
* **Pricing Plans:** Clear pricing tiers and purchase options.
* **Contact Form:** Interactive form to capture visitor inquiries with form validation.
* **Smooth Scrolling:** Seamless navigation between page sections with animated transitions.
* **SEO Optimization:** Proper semantic HTML and meta tags for better search engine ranking.
* **Cross-Browser Compatibility:** Tested on Chrome, Firefox, Safari, and Edge.

**Option 3: Day Planner Website**

This web app helps users organize their day by scheduling tasks in hourly time blocks, improving productivity and time management.

**Key Features:**

* **Time-Block Scheduling:** Users can add tasks in time slots (e.g., 9 AM to 5 PM).
* **Color-Coded Time Blocks:** Past hours are greyed out, current hour highlighted in red, future hours green.
* **Task Persistence:** All tasks are saved in localStorage so that they remain after page reloads.
* **Edit/Delete Tasks:** Modify or remove tasks as needed with user-friendly UI.
* **Responsive Layout:** The planner adjusts gracefully on mobile and desktop.
* **Notification Reminders:** Optional feature to alert users of upcoming tasks (can be browser alerts or UI notifications).
* **Clear All Button:** Allows users to quickly clear the entire day’s schedule.

**4. Technologies Used**

* **HTML5:** Provided the semantic structure and accessibility for web content.
* **CSS3:** Styled the application, including Flexbox/Grid for layouts, media queries for responsiveness, and animations for better user engagement.
* **JavaScript (ES6+):** Implemented DOM manipulation, event handling, form validation, data storage (localStorage), and interactive elements.
* **Libraries (optional):**
  + Chart.js (for Expense Tracker charts)
  + Bootstrap or Tailwind CSS (for faster UI development)
  + jQuery (optional for easier DOM handling)

**5. Development Process**

* **Requirement Analysis:** Understanding the needs and defining clear functional and non-functional requirements.
* **Wireframing & UI Design:** Created low-fidelity sketches followed by high-fidelity mockups to plan the layout and user interface.
* **Implementation:**
  + Structured HTML pages with semantic tags.
  + Styled components for visual appeal and responsiveness.
  + Developed JavaScript functions for dynamic behavior and data persistence.
* **Testing:** Conducted manual and cross-browser testing to ensure functionality and consistent appearance.
* **Feedback & Iteration:** Incorporated suggestions from mentors or users to improve usability and fix bugs.

**Expense/Money Tracker Web Application — Complete Code**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8" />**

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

**<title>Expense/Money Tracker</title>**

**<style>**

**body {**

**font-family: Arial, sans-serif;**

**background: #f7f9fc;**

**margin: 0;**

**padding: 20px;**

**display: flex;**

**justify-content: center;**

**}**

**.container {**

**max-width: 600px;**

**background: white;**

**padding: 20px 30px;**

**border-radius: 8px;**

**box-shadow: 0 2px 10px rgb(0 0 0 / 0.1);**

**}**

**h1 {**

**text-align: center;**

**margin-bottom: 20px;**

**color: #333;**

**}**

**.balance {**

**font-size: 1.5rem;**

**text-align: center;**

**margin-bottom: 20px;**

**}**

**.summary {**

**display: flex;**

**justify-content: space-around;**

**margin-bottom: 20px;**

**}**

**.summary div {**

**background: #eef3f7;**

**padding: 15px 20px;**

**border-radius: 6px;**

**width: 30%;**

**text-align: center;**

**font-weight: bold;**

**color: #555;**

**}**

**form {**

**display: flex;**

**gap: 10px;**

**margin-bottom: 20px;**

**}**

**input, select {**

**padding: 10px;**

**font-size: 1rem;**

**border: 1px solid #ccc;**

**border-radius: 4px;**

**flex: 1;**

**}**

**button {**

**padding: 10px 20px;**

**font-size: 1rem;**

**border: none;**

**border-radius: 4px;**

**background: #007bff;**

**color: white;**

**cursor: pointer;**

**transition: background-color 0.3s ease;**

**}**

**button:hover {**

**background: #0056b3;**

**}**

**ul {**

**list-style-type: none;**

**padding: 0;**

**}**

**li {**

**background: #f1f5f9;**

**padding: 10px 15px;**

**margin-bottom: 10px;**

**border-radius: 4px;**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**font-weight: 500;**

**color: #333;**

**}**

**li.expense {**

**border-left: 5px solid #e55353;**

**}**

**li.income {**

**border-left: 5px solid #53a653;**

**}**

**.delete-btn {**

**background: transparent;**

**border: none;**

**font-size: 1.2rem;**

**color: #999;**

**cursor: pointer;**

**}**

**.delete-btn:hover {**

**color: #e55353;**

**}**

**@media (max-width: 600px) {**

**.summary {**

**flex-direction: column;**

**gap: 10px;**

**}**

**.summary div {**

**width: 100%;**

**}**

**form {**

**flex-direction: column;**

**}**

**button {**

**width: 100%;**

**}**

**}**

**</style>**

**</head>**

**<body>**

**<div class="container">**

**<h1>Expense/Money Tracker</h1>**

**<div class="balance" id="balance">Balance: ₹0</div>**

**<div class="summary">**

**<div>Income: ₹<span id="money-plus">0</span></div>**

**<div>Expenses: ₹<span id="money-minus">0</span></div>**

**<div>Total: ₹<span id="total">0</span></div>**

**</div>**

**<form id="transaction-form">**

**<select id="type" required>**

**<option value="income">Income</option>**

**<option value="expense">Expense</option>**

**</select>**

**<input**

**type="text"**

**id="description"**

**placeholder="Description"**

**required**

**maxlength="50"**

**/>**

**<input**

**type="number"**

**id="amount"**

**placeholder="Amount"**

**required**

**min="0.01"**

**step="0.01"**

**/>**

**<button type="submit">Add</button>**

**</form>**

**<ul id="transaction-list"></ul>**

**</div>**

**<script>**

**// Elements**

**const balanceEl = document.getElementById("balance");**

**const moneyPlusEl = document.getElementById("money-plus");**

**const moneyMinusEl = document.getElementById("money-minus");**

**const totalEl = document.getElementById("total");**

**const transactionList = document.getElementById("transaction-list");**

**const form = document.getElementById("transaction-form");**

**const typeEl = document.getElementById("type");**

**const descriptionEl = document.getElementById("description");**

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

**// Get transactions from localStorage or empty array**

**let transactions =**

**JSON.parse(localStorage.getItem("transactions")) || [];**

**// Add transaction to DOM**

**function addTransactionDOM(transaction) {**

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

**li.classList.add(transaction.type === "expense" ? "expense" : "income");**

**li.innerHTML = `**

**${transaction.description} <span>₹${transaction.amount.toFixed(2)}</span>**

**<button class="delete-btn" data-id="${transaction.id}">&times;</button>**

**`;**

**transactionList.appendChild(li);**

**}**

**// Update summary and balance**

**function updateValues() {**

**const amounts = transactions.map((t) =>**

**t.type === "expense" ? -t.amount : t.amount**

**);**

**const income = transactions**

**.filter((t) => t.type === "income")**

**.reduce((acc, t) => acc + t.amount, 0);**

**const expense = transactions**

**.filter((t) => t.type === "expense")**

**.reduce((acc, t) => acc + t.amount, 0);**

**const total = income - expense;**

**moneyPlusEl.innerText = income.toFixed(2);**

**moneyMinusEl.innerText = expense.toFixed(2);**

**totalEl.innerText = total.toFixed(2);**

**balanceEl.innerText = `Balance: ₹${total.toFixed(2)}`;**

**balanceEl.style.color = total >= 0 ? "green" : "red";**

**}**

**// Remove transaction by ID**

**function removeTransaction(id) {**

**transactions = transactions.filter((t) => t.id !== id);**

**updateLocalStorage();**

**init();**

**}**

**// Save transactions to localStorage**

**function updateLocalStorage() {**

**localStorage.setItem("transactions", JSON.stringify(transactions));**

**}**

**// Init app**

**function init() {**

**transactionList.innerHTML = "";**

**transactions.forEach(addTransactionDOM);**

**updateValues();**

**}**

**// Form submit handler**

**form.addEventListener("submit", (e) => {**

**e.preventDefault();**

**const type = typeEl.value;**

**const description = descriptionEl.value.trim();**

**const amount = parseFloat(amountEl.value);**

**if (description === "" || isNaN(amount) || amount <= 0) {**

**alert("Please enter valid description and amount.");**

**return;**

**}**

**const transaction = {**

**id: Date.now(),**

**type,**

**description,**

**amount,**

**};**

**transactions.push(transaction);**

**updateLocalStorage();**

**init();**

**form.reset();**

**typeEl.focus();**

**});**

**// Delete button handler (event delegation)**

**transactionList.addEventListener("click", (e) => {**

**if (e.target.classList.contains("delete-btn")) {**

**const id = Number(e.target.getAttribute("data-id"));**

**removeTransaction(id);**

**}**

**});**

**// Start app**

**init();**

**</script>**

**</body>**

**</html>**

**6. Challenges Faced and Solutions**

* **Responsive Design Complexity:** Ensured consistent look and feel across various devices by extensively using media queries and flexible layouts.
* **Managing State & Data Persistence:** Handling CRUD operations with localStorage required careful structuring to avoid data loss and maintain UI synchronization.
* **User Input Validation:** Implemented thorough checks to prevent invalid or incomplete data entries.
* **Performance Optimization:** Minimized unnecessary DOM updates and optimized JavaScript to keep the app fast.
* **Cross-Browser Bugs:** Used developer tools to debug and fix CSS/JS compatibility issues on different browsers.

**7. Learning Outcomes**

* Enhanced proficiency in front-end web technologies (HTML, CSS, JavaScript).
* Gained practical experience in building responsive and interactive web applications.
* Learned to manage client-side data using localStorage effectively.
* Developed skills in UI/UX design principles for better user engagement.
* Improved problem-solving skills by debugging and optimizing code.
* Understood the software development lifecycle from planning to deployment.

**8. Conclusion**

This internship project provided an invaluable opportunity to apply theoretical knowledge to real-world web development scenarios. Whether it was developing a financial tracker, creating an attractive landing page, or building an efficient day planner, I gained hands-on experience that significantly improved my technical and soft skills. The project not only enhanced my coding abilities but also taught me the importance of clean design, usability, and responsiveness — all critical components of modern web applications.