**ENERGY STUDIO**

**SANTHOSHI.B**

**SHABANA.K**

**SANDHIYA** **SREE.J**

**SEVANTHI.A**

**ABSTRACT**:

A Energy studioApplication is a web-based platform developed using HTML, CSS, and JavaScript to help users monitor their fitness journey efficiently. The application allows users to schedule workouts, track calorie burn, log water intake,Heart rate monitor,calorie calculator ,Progress & Achievements **,**Task Manager and review workout history with an interactive and user-friendly dark-themed UI. It features a login/signup system for personalized tracking, ensuring data security. The dashboard presents workout schedules, calorie charts, water intake statistics, and progress highlights using dynamic charts and interactive elements. heart rate monitor that tracks BPM (beats per minute) and calculates calories burned based on height, weight, and age. progress tracking, achievements, and step counting. Task Manager that includes pending and completed tasks trackingTime Breaker that includes a Clock, Stopwatch, Timer, and Basic Warm-up Exercises. With a neon-themed interface and moving particle effects, the Energy studio provides an engaging experience while helping users maintain a healthy lifestyle.

**SYNOPSIS :**

**Title: Energy Studio Application Using HTML, CSS, and JavaScript**

**Introduction:**

In today’s fast-paced world, maintaining a **healthy lifestyle** is a challenge, and many individuals struggle to keep track of their fitness progress. A **Energy studioApplication** can serve as an effective tool to help users **monitor their workouts, track calorie intake, and maintain a healthy routine**. This project aims to develop a **web-based energy studio** using **HTML, CSS, and JavaScript**, providing users with a structured and interactive platform to manage their fitness activities.

The application features a **dark-themed UI with neon effects**, making it visually appealing and easy to navigate. Users can **schedule their workouts, log water intake, track calories burned, and view workout history** in an organized manner. The use of **charts and interactive elements** enhances user engagement and provides clear insights into their fitness journey. Additionally, the application includes **step tracking, Health monitor,calorie calculator, Progress & Achievements,Task Manager,time breaker with clock,timer,stop watch** and also the **basic warm-up exersice** to encourage users to stay motivated.

By implementing a **secure login/signup system**, users can create personal accounts and store their fitness data securely. The fitness tracker ensures a **personalized and user-friendly experience**, making it a valuable tool for fitness enthusiasts who want to monitor and improve their health effectively.

**Module Description –Energy studio Application :**

The **Energy Studio Application** is divided into several modules, each responsible for specific functionalities to ensure a seamless user experience. Below are the key modules of the application:

**Objective:**

* To develop an easy-to-use Energy studio web application.
* To allow users to **schedule workouts with specific details** such as time, day, and calories burned.
* To provide a **calorie tracking system** with pie charts for better visualization.
* To enable users to **log water intake** and display it using a **pie graph.**
* **heart rate monitor** that tracks **BPM (beats per minute).**
* calculates **calories burned** based on height, weight, and age.
* **Step counting** with distance and activity tracking
* **Task Manager helps users organize tasks by categorizing them as pending or completed. Users can add, edit, delete, and track their progress efficiently.**
* The **Time Breaker** is a time management and workout tool.
* To offer a **secure login/signup system** with saved credentials for user authentication.

**Scope:**

This application is designed for **fitness enthusiasts** who want to track their workout routines and health progress in a structured manner. It provides a **personalized dashboard** where users can **manage their fitness data, track progress, and stay motivated**. The use of **charts, animations, and interactive UI elements** makes it more engaging and efficient.

**Features:**

1. **User Authentication:** Signup and login functionality with secure credential storage.
2. **Workout Scheduling:** Users can add workouts with time, day, and calories burned.
3. **Workout History:** Users can view, edit, and manage past workout logs.
4. **Calorie Tracking:** calorie burn trends for the week.
5. **Water Intake Logging:** Users can input daily water intake, represented using a **pie graph.**
6. **Heart beat monitor:** **Real-time heart rate (BPM) tracking** uses the PPG (photoplethysmography) or ECG (electrocardiogram) to detect pulse rate.
7. **Calorie calculator:** Calorie calculation using user input (height, weight, age).
8. **Progress & Achievements**: This system tracks a user's **step count**, monitors **progress**, and unlocks **achievements** based on milestones.
9. **Task Manager**: Add tasks with descriptions and due dates Mark tasks as completed, View pending and completed tasks separately.

10.**Dark-Themed UI:** A visually appealing **neon theme.**

**1. User Authentication Module**

**Purpose:**

* Allows users to **sign up and log in** securely.
* Stores user credentials using **local storage** for authentication.
* Prevents unauthorized access and ensures data privacy.

**Features:**

* **Signup Page:** Users can create an account by entering their **First Name, Last Name, Phone Number, Email, Password, and Confirm Password**.
* **Login Page:** Users can log in with their registered email and password.
* **Credential Validation:** Checks entered details against stored credentials before allowing access.

**2. Dashboard Module**

**Purpose:**

* Acts as the **central hub** where users can view and manage their fitness data.
* Displays **various health metrics** in an organized layout.

**Features:**

* **Workout Scheduling Section:** Users can schedule workouts by selecting **time, workout name, date, and calories burned**.
* **Workout History Section:** Displays past workout logs with an **delete button** for modifications.
* **Calories Burned Chart:** calories burned for the week.
* **Water Intake Pie Chart:** A graphical representation of daily water intake.

**3. Workout Scheduling Module**

**Purpose:**

* Allows users to **add and manage workouts** for better fitness planning.

**Features:**

* Users can input **workout details (time, name, date, calories burned)**.
* Workouts are displayed in the **schedule section** on the dashboard.
* Data is stored using **local storage** for easy access and modification.

**4. Workout History Module**

**Purpose:**

* Keeps a record of **all past workouts**, allowing users to track their progress over time.

**Features:**

* Displays a **scrollable list of previous workouts**.
* Includes an **delete button** to modify workout details.
* Prevents **undefined values** from appearing in the history section.

**5. Calorie Tracking Module**

**Purpose:**

* Helps users visualize their **calories burned** over a period of time.

**Features:**

* A **pie chart** that displays **daily calorie data**.
* Updates dynamically based on **user input** from scheduled workouts.

**6. Water Intake Tracking Module**

**Purpose:**

* Allows users to **log and monitor daily water consumption**.

**Features:**

* Users can **add water intake** while scheduling workouts.
* The **pie graph**  visually represents daily water consumption levels.

**7. UI & Theme Module**

**Purpose:**

* Ensures a **visually appealing and user-friendly interface**.

**Features:**

* **Dark-themed UI** with **neon accents** for a modern look.
* **styled buttons** for signup, login, and navigation.

**System Specifications – Energy studio Application :**

The **Energy Studio Application** is a web-based platform developed using **HTML, CSS, and JavaScript**. The following system specifications define the necessary hardware and software requirements for running and developing the application efficiently.

**1. Hardware Requirements**

**For Development:**

* **Processor:** Intel Core i3 or higher (or AMD equivalent)
* **RAM:** Minimum 4GB (Recommended: 8GB or more)
* **Storage:** At least 10GB of free space
* **Display:** 1366×768 resolution or higher
* **Graphics Card:** Integrated or dedicated GPU (optional for UI design tools)
* **Internet Connection:** Required for development tools and libraries

**For End Users (Minimum Requirements):**

* **Device:** Desktop and laptop.
* **Processor:** Any modern processor (Dual-core or higher)
* **RAM:** 2GB or more
* **Storage:** Minimal, as the application is web-based
* **Internet Browser:** Chrome, Firefox, Edge, or any modern browser
* **Internet Connection:** Required for online usage (if hosted on a server)

**2. Software Requirements**

**For Development:**

* **Operating System:** Windows 10/11, macOS, or Linux
* **Code Editor:** VS Code, Sublime Text, or any preferred text editor
* **Browser:** Google Chrome, Mozilla Firefox, or Microsoft Edge (for testing)

**For End Users:**

* **Operating System:** Windows, macOS, Linux, Android, or iOS
* **Browser Compatibility:**
  + Google Chrome (Latest Version)
  + Mozilla Firefox
  + Microsoft Edge
  + Safari (for macOS/iOS users)

**3. Technology Stack**

**Frontend:**

* + HTML (Structure)
  + CSS (Styling)
  + JavaScript (Functionality & Interactivity)

**Backend:**

* + JavaScript

**Data storage:**

* + Local storage

**UI Enhancements:**

* + CSS Animations
  + JavaScript Interactivity
  + Chart.js (For visualizing data)

**4. Deployment Requirements**

* **Local Deployment:** Can run directly on a browser without a server.
* **Web Hosting (Optional for Online Use):**
  + GitHub Pages

**System Study – Advantages & Disadvantages**

The **Energy Studio Application** is designed to provide users with an efficient way to track their fitness activities. However, like any system, it has both advantages and disadvantages.

**Advantages:**

**1. User-Friendly Interface**

* The application features a **dark-themed UI with neon effects**, making it visually appealing and easy to navigate.
* Interactive elements such as buttons, charts, and animations enhance the user experience.

**2. Easy Workout & Health Tracking**

* Users can **schedule workouts, track calorie burn, and log water intake** easily.
* The **calorie burned pie chart** and **water intake pie graph** provide visual insights into fitness progress.

**3. Web-Based & Cross-Platform Compatibility**

* Since the application is developed using **HTML, CSS, and JavaScript**, it can be accessed from **any device with a web browser**.
* No need for installation, making it lightweight and easy to use.

**4. Secure User Authentication**

* The login/signup system ensures **secure access to personal fitness data**.
* Credentials are stored safely using **local storage** (or can be extended with Firebase for cloud storage).

**5. Data Visualization**

* The application uses **charts and graphs** to represent **calories burned, water intake, and workout history**, helping users analyze their progress better.

**6. Customization & Scalability**

* The application can be **expanded** to include features like **goal setting, diet tracking, and AI-based recommendations** in the future.

**Disadvantages:**

**1. Limited Data Storage**

* Since the application currently relies on **local storage**, users may lose their data if they **clear browser cache or switch devices**.
* No **cloud backup** (unless Firebase or a database is integrated).

**2. No Real-Time Syncing**

* The data does not sync across multiple devices unless an **online database** is implemented.
* Users must use the same device to access their workout history.

**3. No Automated Workout Suggestions**

* The system does not provide **personalized workout recommendations** based on user performance.
* AI or machine learning integration would be needed for **automated fitness advice**.

**4. Dependency on Manual Input**

* Users have to **manually enter workout details, water intake, and calories burned**.
* No integration with **fitness wearables or step counters** for automated tracking.

**System Testing for Energy Studio Application**

System testing ensures that the **Energy studio Application** functions correctly and meets user requirements. It involves different types of testing to identify and fix errors before deployment.

**1. Testing Types**

**A. Functional Testing**

Checks if all features work as expected.  
✅ **Test Cases:**

* **User Authentication** (Login/Signup)
  + Valid credentials → Successful login
  + Invalid credentials → Error message
* **Workout Scheduling**
  + Add workout → Saved successfully
  + Delete workout → Removed from history
* **Water Intake Tracking**
  + Add water intake → Displays correct amount
* **Calorie Tracking & pie charts**
  + Enter calories → Updates pie chart correctly
* **Dashboard & History**
  + Display correct past workouts.
* **Health monitor**
* Heart beat → checks heart beat as per the minutes.
* **Calorie calculator**
* Enter age,weight,height → display calorie.
* **Progress & archivement**
* Add step count → display ranking.
* **Task manager**
* Add task → display the completed task and pending task.
* **Time breaker**
* Clock , stop watch , timer will be display.
* **Basic warm up exercise**
* Display some basic exercises.

**B. User Interface (UI) Testing**

Ensures UI elements work properly and follow design guidelines.  
✅ **Test Cases:**

* Buttons, text fields, and navigation work smoothly
* Dark-themed UI with **neon effect** is properly displayed
* Responsive design adapts to different screen sizes

**C. Performance Testing**

Measures the system’s speed, responsiveness, and stability.  
✅ **Test Cases:**

* Application loads within **2 seconds**
* Handles **multiple users’ data** without lag
* Charts and analytics update instantly

**D. Security Testing**

Verifies that user data is protected.  
✅ **Test Cases:**

* Password encryption works correctly

**E. Compatibility Testing**

Checks if the application works on different devices and browsers.  
✅ **Test Cases:**

* Runs smoothly on **Chrome, Firefox, Edge**
* Functions correctly on **laptop and desktop**

**F. Usability Testing**

Ensures that the app is easy to use.  
✅ **Test Cases:**

* Users can navigate the app easily
* Workout tracking process is intuitive
* Clear error messages for invalid inputs

**2. Test Execution & Bug Fixing**

* Execute test cases **manually or using automation tools**
* Identify and fix bugs
* Retest the system to ensure fixes work

**3. Test Summary Report**

A final report detailing test results, bugs found, and fixes applied.

**SOURCE CODE:**

**DASHBOARD :**

**Html code:**

**<!DOCTYPE html>**

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

**<head>**

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

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

**<title>Fitness Tracker</title>**

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

**<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>**

**</head>**

**<body>**

**<h1>Fitness Tracker Dashboard</h1>**

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

**<!-- Schedule Workout -->**

**<div class="workout-schedule">**

**<h2>Schedule Your Workout</h2>**

**<label for="day">Day:</label>**

**<select id="day">**

**<option value="">Select Day</option>**

**<option value="Monday">Monday</option>**

**<option value="Tuesday">Tuesday</option>**

**<option value="Wednesday">Wednesday</option>**

**<option value="Thursday">Thursday</option>**

**<option value="Friday">Friday</option>**

**<option value="Saturday">Saturday</option>**

**<option value="Sunday">Sunday</option>**

**</select>**

**<label for="workout">Workout Name:</label>**

**<input type="text" id="workout" placeholder="Enter workout">**

**<label for="time">Time:</label>**

**<input type="time" id="time">**

**<label for="calories">Calories Burned:</label>**

**<input type="number" id="calories" placeholder="Enter calories (optional)">**

**<label for="water">Water Intake (ml):</label>**

**<input type="number" id="water" placeholder="Enter water intake">**

**<button onclick="addWorkout()">Add Workout</button>**

**</div>**

**<!-- Workout History -->**

**<div class="workout-history">**

**<h2>Workout History</h2>**

**<div class="history-list" id="history">**

**<!-- History entries will appear here -->**

**</div>**

**<button onclick="clearHistory()">Clear History</button>**

**</div>**

**<!-- Calories Burned Chart -->**

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

**<h2>Calories Burned Chart</h2>**

**<canvas id="caloriesChart"></canvas>**

**</div>**

**<!-- Water Intake Pie Chart -->**

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

**<h2>Water Intake Pie Chart</h2>**

**<canvas id="waterChart"></canvas>**

**</div>**

**</div>**

**<!-- Next Page Button -->**

**<div class="next-page-container">**

**<a href="health.html" class="next-page-btn">**

**<img src="ay.jpg" alt="Next Page" class="next-icon">**

**</a>**

**</div>**

**<script src="script.js"></script>**

**</body>**

**</html>**

**Script js:**

**let historyData = JSON.parse(localStorage.getItem("historyData")) || [];  // Load stored history**

**let caloriesData = { Monday: 0, Tuesday: 0, Wednesday: 0, Thursday: 0, Friday: 0, Saturday: 0, Sunday: 0 };**

**let waterData = {};**

**// Chart Variables**

**let caloriesChartInstance;**

**let waterChartInstance;**

**function addWorkout() {**

**let day = document.getElementById("day").value;**

**let workout = document.getElementById("workout").value;**

**let time = document.getElementById("time").value;**

**let calories = document.getElementById("calories").value;**

**let water = document.getElementById("water").value;**

**if (!day || !workout || !time) {**

**alert("Please fill in at least Day, Workout, and Time.");**

**return;**

**}**

**let entry = { day, workout, time, calories: calories || 0, water: water || 0 };**

**historyData.push(entry);**

**saveData();  // Save updated history to storage**

**updateHistory();**

**updateCharts();**

**}**

**function saveData() {**

**localStorage.setItem("historyData", JSON.stringify(historyData));  // Save history to localStorage**

**}**

**function updateHistory() {**

**let historyDiv = document.getElementById("history");**

**historyDiv.innerHTML = "";**

**historyData.forEach((entry, index) => {**

**let div = document.createElement("div");**

**div.classList.add("history-entry");**

**div.innerHTML = `**

**<p><strong>${entry.day}:</strong> ${entry.workout} at ${entry.time}, ${entry.calories} cal</p>**

**<div class="button-group">**

**<button class="edit-btn" onclick="editWorkout(${index})">Edit</button>**

**<button class="delete-btn" onclick="deleteWorkout(${index})">Delete</button>**

**</div>**

**`;**

**historyDiv.appendChild(div);**

**});**

**}**

**// Function to delete a workout entry**

**function deleteWorkout(index) {**

**if (confirm("Are you sure you want to delete this workout?")) {**

**historyData.splice(index, 1);**

**saveData();**

**updateHistory();**

**}**

**}**

**function editWorkout(index) {**

**let entry = historyData[index];**

**document.getElementById("day").value = entry.day;**

**document.getElementById("workout").value = entry.workout;**

**document.getElementById("time").value = entry.time;**

**document.getElementById("calories").value = entry.calories;**

**document.getElementById("water").value = entry.water;**

**historyData.splice(index, 1);**

**saveData();**

**updateHistory();**

**}**

**function editWorkout(index) {**

**let entry = historyData[index];**

**document.getElementById("day").value = entry.day;**

**document.getElementById("workout").value = entry.workout;**

**document.getElementById("time").value = entry.time;**

**document.getElementById("calories").value = entry.calories;**

**document.getElementById("water").value = entry.water;**

**historyData.splice(index, 1);**

**saveData();  // Save changes to localStorage**

**updateHistory();**

**updateCharts();**

**}**

**function clearHistory() {**

**historyData = [];**

**localStorage.removeItem("historyData");  // Clear stored history**

**updateHistory();**

**updateCharts();**

**}**

**function updateCharts() {**

**caloriesData = { Monday: 0, Tuesday: 0, Wednesday: 0, Thursday: 0, Friday: 0, Saturday: 0, Sunday: 0 };**

**waterData = {};**

**historyData.forEach(entry => {**

**caloriesData[entry.day] += Number(entry.calories);**

**waterData[entry.day] = (waterData[entry.day] || 0) + Number(entry.water);**

**});**

**drawCaloriesChart();**

**drawWaterChart();**

**}**

**function drawCaloriesChart() {**

**if (caloriesChartInstance) caloriesChartInstance.destroy();**

**let ctx = document.getElementById("caloriesChart").getContext("2d");**

**let neonColors = ["#00FF7F", "#00E5FF", "#FF007F", "#9400D3", "#00FF7F", "#00E5FF", "#FF007F"];**

**caloriesChartInstance = new Chart(ctx, {**

**type: "bar",**

**data: {**

**labels: Object.keys(caloriesData),**

**datasets: [{**

**label: "Calories Burned",**

**data: Object.values(caloriesData),**

**backgroundColor: neonColors,**

**borderColor: "#FFF",**

**borderWidth: 1,**

**hoverBackgroundColor: "#FFFF00", // Yellow glow on hover**

**}]**

**},**

**options: {**

**responsive: true,**

**plugins: {**

**legend: { labels: { color: "#E0E0E0" } }**

**},**

**scales: {**

**x: { ticks: { color: "#E0E0E0" }, grid: { color: "#555" } },**

**y: { ticks: { color: "#E0E0E0" }, grid: { color: "#555" } }**

**}**

**}**

**});**

**}**

**function drawWaterChart() {**

**if (waterChartInstance) waterChartInstance.destroy();**

**let ctx = document.getElementById("waterChart").getContext("2d");**

**let pieColors = ["#00E5FF", "#FF007F", "#9400D3", "#00FF7F", "#FF6B6B", "#FFD700", "#34D399"];**

**waterChartInstance = new Chart(ctx, {**

**type: "pie",**

**data: {**

**labels: Object.keys(waterData),**

**datasets: [{**

**data: Object.values(waterData),**

**backgroundColor: pieColors,**

**borderColor: "#FFF",**

**hoverBackgroundColor: "#FFFF00", // Yellow glow on hover**

**borderWidth: 1**

**}]**

**},**

**options: {**

**responsive: true,**

**plugins: {**

**legend: { labels: { color: "#E0E0E0" } }**

**}**

**}**

**});**

**}**

**// Load stored history on page load**

**document.addEventListener("DOMContentLoaded", function () {**

**historyData = JSON.parse(localStorage.getItem("historyData")) || [];  // Load history**

**updateHistory();**

**updateCharts();**

**});**

**CSS:**

**@import url('https://fonts.googleapis.com/css2?family=Rajdhani:wght@400;700&display=swap');**

**body {**

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

**background-color: #121212; /\* Dark background \*/**

**margin: 0;**

**padding: 20px;**

**text-align: center;**

**color: #E0E0E0; /\* Soft white text \*/**

**}**

**/\* Layout Containers \*/**

**.container {**

**display: grid;**

**grid-template-columns: 1fr 1fr;**

**grid-template-rows: auto auto;**

**gap: 20px;**

**width: 80%;**

**margin: auto;**

**}**

**.left-container, .right-container {**

**width: 48%;**

**height: 90vh;**

**padding: 20px;**

**background: #1E1E1E; /\* Dark gray \*/**

**border-radius: 10px;**

**box-shadow: 0px 0px 10px rgba(0, 255, 127, 0.2);**

**overflow: auto;**

**}**

**/\* Cards & Sections \*/**

**.workout-schedule, .workout-history, .chart-container {**

**background: #1A1A1A; /\* Darker section background \*/**

**padding: 20px;**

**border-radius: 10px;**

**box-shadow: 0 0 15px rgba(0, 255, 127, 0.2); /\* Neon glow \*/**

**border: 1px solid #00FF7F;**

**transition: transform 0.2s ease, box-shadow 0.3s ease;**

**}**

**.workout-schedule:hover, .workout-history:hover, .chart-container:hover {**

**transform: scale(1.02);**

**box-shadow: 0 0 20px rgba(0, 255, 127, 0.5);**

**}**

**/\* Time Display \*/**

**.time-display {**

**font-size: 2em;**

**font-weight: bold;**

**color: #00FF7F;**

**}**

**/\* Buttons \*/**

**button {**

**display: block;**

**margin-top: 10px;**

**padding: 12px;**

**background-color: #00FF7F;**

**color: #121212;**

**border: none;**

**cursor: pointer;**

**font-weight: bold;**

**border-radius: 8px;**

**box-shadow: 0 0 10px rgba(0, 255, 127, 0.8);**

**transition: transform 0.2s ease, box-shadow 0.3s ease;**

**}**

**button:hover {**

**background-color: #00cc66;**

**box-shadow: 0 0 15px rgba(0, 255, 127, 1);**

**transform: scale(1.05);**

**}**

**/\* Inputs & Selects \*/**

**select, input {**

**width: 100%;**

**padding: 12px;**

**margin-top: 10px;**

**background: #2A2A2A;**

**color: #E0E0E0;**

**border: 1px solid #00FF7F;**

**border-radius: 8px;**

**transition: border 0.3s ease, box-shadow 0.3s ease;**

**}**

**select option {**

**background: #1A1A1A;**

**color: #E0E0E0;**

**}**

**input:focus, select:focus {**

**border-color: #00cc66;**

**outline: none;**

**box-shadow: 0 0 10px rgba(0, 255, 127, 0.8);**

**}**

**/\* Workout History Entries \*/**

**.history-entry {**

**background: #2A2A2A;**

**padding: 10px;**

**margin: 5px 0;**

**border-radius: 5px;**

**color: #E0E0E0;**

**display: flex;**

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

**align-items: center;**

**transition: transform 0.2s ease, box-shadow 0.3s ease;**

**}**

**.history-entry:hover {**

**transform: scale(1.02);**

**box-shadow: 0 0 10px rgba(0, 255, 127, 0.6);**

**}**

**/\* Button Group \*/**

**.button-group {**

**display: flex;**

**gap: 10px;**

**}**

**/\* Edit and Delete Buttons \*/**

**.edit-btn, .delete-btn {**

**padding: 6px 12px;**

**font-size: 14px;**

**font-weight: bold;**

**border: none;**

**border-radius: 6px;**

**cursor: pointer;**

**transition: transform 0.2s ease, box-shadow 0.3s ease;**

**}**

**/\* Edit Button \*/**

**.edit-btn {**

**background: linear-gradient(135deg, #007BFF, #0056b3);**

**color: white;**

**box-shadow: 0 0 10px rgba(0, 123, 255, 0.8);**

**}**

**.edit-btn:hover {**

**background: linear-gradient(135deg, #0056b3, #003d80);**

**box-shadow: 0 0 15px rgba(0, 123, 255, 1);**

**transform: scale(1.05);**

**}**

**/\* Delete Button \*/**

**.delete-btn {**

**background: linear-gradient(135deg, #FF4C4C, #CC0000);**

**color: white;**

**box-shadow: 0 0 10px rgba(255, 76, 76, 0.8);**

**}**

**.delete-btn:hover {**

**background: linear-gradient(135deg, #CC0000, #990000);**

**box-shadow: 0 0 15px rgba(255, 76, 76, 1);**

**transform: scale(1.05);**

**}**

**/\* Next Page Button \*/**

**.next-page-container {**

**position: absolute;**

**top: 20px;**

**right: 20px;**

**}**

**.next-page-btn {**

**display: inline-block;**

**background: rgba(0, 255, 127, 0.2);**

**padding: 10px;**

**border-radius: 50%;**

**box-shadow: 0 0 8px rgba(0, 255, 127, 0.5);**

**transition: transform 0.3s ease, box-shadow 0.3s ease;**

**}**

**.next-page-btn:hover {**

**transform: scale(1.1);**

**box-shadow: 0 0 12px rgba(0, 255, 127, 1);**

**}**

**/\* Back Page Button \*/**

**.back-page-container {**

**position: absolute;**

**top: 20px;**

**left: 20px;**

**}**

**.back-page-btn {**

**display: inline-block;**

**background: rgba(0, 255, 127, 0.2);**

**padding: 10px;**

**border-radius: 50%;**

**box-shadow: 0 0 8px rgba(0, 255, 127, 0.5);**

**transition: transform 0.3s ease, box-shadow 0.3s ease;**

**}**

**.back-page-btn:hover {**

**transform: scale(1.1);**

**box-shadow: 0 0 12px rgba(0, 255, 127, 1);**

**}**

**/\* Fix Arrow Icon Size \*/**

**.next-icon, .back-icon {**

**width: 32px;**

**height: 32px;**

**}**

**/\* Responsive Design \*/**

**@media screen and (max-width: 768px) {**

**.container {**

**grid-template-columns: 1fr;**

**width: 90%;**

**}**

**canvas {**

**height: 220px !important;**

**}**

**.next-page-container, .back-page-container {**

**top: 10px;**

**}**

**.next-icon, .back-icon {**

**width: 28px;**

**height: 28px;**

**}**

**}**

**HEALTH MONITOR:**

**Html:**

**<!DOCTYPE html>**

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

**<head>**

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

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

**<title>Health Monitor</title>**

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

**<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>**

**</head>**

**<body>**

**<!-- Back Button (Top-Left) -->**

**<div class="back-page-container">**

**<a href="dashboard.html" class="back-page-btn">**

**<img src="dd.jpg" alt="Back" class="back-icon">**

**</a>**

**</div>**

**<!-- Next Page Button (Top-Right) -->**

**<div class="next-page-container">**

**<a href="progress.html" class="next-page-btn">**

**<img src="ay.jpg" alt="Next Page" class="next-icon">**

**</a>**

**</div>**

**<h1>Health Monitor</h1>**

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

**<!-- Heart Rate Tracker -->**

**<div class="workout-schedule">**

**<h2>Heart Rate Monitor</h2>**

**<p>Heart Rate: <span id="bpm">--</span> BPM</p>**

**<button onclick="trackHeartbeat()">Check Heart Rate</button>**

**</div>**

**<!-- Calories Calculator -->**

**<div class="workout-schedule">**

**<h2>Calories Calculator</h2>**

**<label for="age">Age:</label>**

**<input type="number" id="age" placeholder="Enter age">**

**<label for="height">Height (cm):</label>**

**<input type="number" id="height" placeholder="Enter height">**

**<label for="weight">Weight (kg):</label>**

**<input type="number" id="weight" placeholder="Enter weight">**

**<button onclick="calculateCalories()">Calculate Calories</button>**

**<p>Estimated Calories: <span id="caloriesResult">--</span> kcal</p>**

**</div>**

**</div>**

**<script src="health.js"></script>**

**</body>**

**</html>**

**Script js:**

**// Function to Simulate Heart Rate (Random BPM)**

**function trackHeartbeat() {**

**let bpm = Math.floor(Math.random() \* (120 - 60) + 60); // Random BPM between 60-120**

**document.getElementById("bpm").innerText = bpm;**

**}**

**// Function to Calculate Calories Based on Height, Weight & Age**

**function calculateCalories() {**

**let age = document.getElementById("age").value;**

**let height = document.getElementById("height").value;**

**let weight = document.getElementById("weight").value;**

**if (age && height && weight) {**

**let calories = Math.round((10 \* weight) + (6.25 \* height) - (5 \* age) + 5); // Basic calorie formula**

**document.getElementById("caloriesResult").innerText = calories;**

**} else {**

**alert("Please enter all fields.");**

**}**

**CALENDER TODO :**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Calendar and Workout Tracker</title>

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<style>

body {

font-family: 'Poppins', sans-serif;

display: flex;

flex-direction: column;

align-items: center;

height: 100vh;

background: url('pexels-photo-4327012.jpg') no-repeat center center/cover;

color: white;

padding: 10px;

}

.container {

display: flex;

gap: 30px;

flex-wrap: wrap;

justify-content: center;

margin-top: 30px;

width: 80%;

}

.calendar-container, .todo-container {

background: rgba(0, 0, 0, 0.9);

padding: 10px;

border-radius: 15px;

box-shadow: 0 0 10px #00ffcc;

text-align: center;

width: 70%;

transition: 0.3s;

}

.todo-container {

width: 800%; /\* Adjust as needed \*/

max-width: 1000px; /\* Set a max width limit \*/

}

#workoutInput {

width: 300%; /\* Makes it stretch within the container \*/

max-width: 300px; /\* Limit input field size \*/

}

#workoutList {

max-height: 100px; /\* Prevents overflow if too many items \*/

overflow-y: auto; /\* Adds scroll if necessary \*/

}

.calendar {

width: 300%; /\* Makes it stretch within its container \*/

max-width: 300px; /\* Set a limit \*/

}

.calendar-container:hover, .todo-container:hover {

box-shadow: 0 0 15px #00ffcc;

}

.chart-container {

display: block;

margin-top: 15px;

width: 55%;

background: black;

padding: 15px;

border-radius: 15px;

box-shadow: 0 0 10px #ff00ff;

text-align: center;

transition: 0.3s;

}

canvas {

max-width: 650px;

max-height: 650px;

margin: auto;

}

h1, h3 {

font-size: 1.8em; /\* Slight increase in font size \*/

}

input, button {

font-size: 1.2em; /\* Slight increase for better readability \*/

}

::placeholder {

font-size: 1.1em; /\* Increase placeholder text size \*/

}

input[type="date"] {

font-size: 1.2em; /\* Increase date input text size \*/

padding: 5px;

}

</style>

</head>

<body>

<h1>📅 CALENDAR & 🏋️ WORKOUT TRACKER</h1>

<div class="container">

<div class="calendar-container">

<h3>📅 Calendar</h3>

<input type="date" id="workoutDate" class="calendar">

</div>

<div class="todo-container">

<h3>🏋️ Completed Workouts</h3>

<input type="text" id="workoutInput" placeholder="Add a workout">

<button onclick="addWorkout()">Add</button>

<ul id="workoutList"></ul>

</div>

</div>

<div class="chart-container" id="chartContainer">

<h3>📊 Workout Overview</h3>

<canvas id="workoutChart"></canvas>

</div>

<script>

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

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

let colorPalette = ["#FFD700", "#36A2EB", "#FF6384", "#4BC0C0", "#9966FF", "#FF9F40"];

let colorIndex = Object.keys(colors).length;

function addWorkout() {

let input = document.getElementById("workoutInput");

let dateInput = document.getElementById("workoutDate").value;

let workout = input.value.trim();

let dateTime = dateInput ? `${dateInput} ${new Date().toLocaleTimeString()}` : new Date().toLocaleString();

if (workout === "") return;

let ul = document.getElementById("workoutList");

let li = document.createElement("li");

li.innerHTML = `${workout} <span style='font-size: 0.8em; color: gray;'>(${dateTime})</span> <span class='delete' onclick='removeWorkout(this, "${workout}", "${dateTime}")'>✖</span>`;

ul.appendChild(li);

if (!Array.isArray(workoutData[workout])) {

workoutData[workout] = [];

}

workoutData[workout].push(dateTime);

if (!colors[workout]) {

colors[workout] = colorPalette[colorIndex % colorPalette.length];

colorIndex++;

}

localStorage.setItem('workouts', JSON.stringify(workoutData));

localStorage.setItem('workoutColors', JSON.stringify(colors));

updateChart();

input.value = "";

}

function removeWorkout(element, workout, dateTime) {

element.parentElement.remove();

if (Array.isArray(workoutData[workout])) {

workoutData[workout] = workoutData[workout].filter(entry => entry !== dateTime);

if (workoutData[workout].length === 0) delete workoutData[workout];

}

localStorage.setItem('workouts', JSON.stringify(workoutData));

updateChart();

}

function updateChart() {

let ctx = document.getElementById("workoutChart").getContext("2d");

if (window.workoutChartInstance) window.workoutChartInstance.destroy();

window.workoutChartInstance = new Chart(ctx, {

type: "pie",

data: {

labels: Object.keys(workoutData),

datasets: [{

data: Object.keys(workoutData).map(workout => workoutData[workout].length),

backgroundColor: Object.keys(workoutData).map(workout => colors[workout] || "#ddd"),

}],

},

options: {

plugins: {

legend: {

position: 'right',

labels: {

color: 'white',

}

}

}

}

});

}

function loadWorkouts() {

let ul = document.getElementById("workoutList");

for (let workout in workoutData) {

if (Array.isArray(workoutData[workout])) {

workoutData[workout].forEach(dateTime => {

let li = document.createElement("li");

li.innerHTML = `${workout} <span style='font-size: 0.8em; color: gray;'>(${dateTime})</span> <span class='delete' onclick='removeWorkout(this, "${workout}", "${dateTime}")'>✖</span>`;

ul.appendChild(li);

});

}

}

updateChart();

}

window.onload = loadWorkouts;

</script>

</body>

</html>

**TIMER AND STOP WATCH :**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Clock & Workout</title>

<style>

@import url('https://fonts.googleapis.com/css2?family=Rajdhani:wght@400;700&display=swap');

body {

font-family: 'Rajdhani', sans-serif;

display: flex;

justify-content: space-between;

align-items: center;

height: 100vh;

background: url('pexels-photo-1522285.jpg') no-repeat center center fixed;

background-size: cover;

color: white;

margin: 0;

padding: 20px;

animation: fadeIn 1.5s ease-in-out;

}

@keyframes fadeIn {

from { opacity: 0; }

to { opacity: 1; }

}

.left-container, .right-container {

width: 48%;

height: 90vh;

padding: 20px;

background: rgba(30, 30, 30, 0.7);

border-radius: 10px;

box-shadow: 0px 0px 10px cyan;

overflow: auto;

}

.left-container {

animation: glowEffect 3s infinite alternate;

}

@keyframes glowEffect {

from { box-shadow: 0px 0px 10px cyan; }

to { box-shadow: 0px 0px 10px cyan; }

}

.container {

text-align: center;

background: rgba(30, 30, 30, 0.8);

padding: 15px;

border-radius: 10px;

box-shadow: 0px 0px 10px #00ffcc;

margin: 10px 0;

border: 1.5px solid #00ffcc;

}

.time-display {

font-size: 1.9em;

margin: 10px 0;

font-weight: bold;

color: #00ffcc;

}

button {

margin: 3px;

padding: 8px 15px;

border: none;

background: #00ffcc;

color: #121212;

cursor: pointer;

border-radius: 6px;

font-size: 0.9em;

font-weight: bold;

transition: transform 0.3s;

}

button:hover {

transform: scale(1.1);

}

.exercise-container {

display: grid;

grid-template-columns: repeat(3, 1fr);

gap: 15px;

justify-content: center;

}

.modal-content img {

width: 300px; /\* Adjust as needed \*/

height: 300px; /\* Adjust as needed \*/

object-fit: cover; /\* Ensures the image scales properly \*/

border-radius: 8px;

}

.exercise {

width: 75px;

cursor: pointer;

text-align: center;

padding: 6px;

background: rgba(255, 255, 255, 0.1);

border-radius: 6px;

border: 2px solid cyan;

box-shadow: 0 0 6px cyan;

transition: transform 0.3s, box-shadow 0.3s;

}

.exercise:hover {

transform: scale(1.1);

box-shadow: 0 0 6px cyan;

}

.exercise img {

width: 100%;

border-radius: 4px;

}

</style>

<div class="content">

<button class=" MOVE TO CALENDAR">

<a href="calendar todo.html" class="btn"> <<< MOVE TO CALENDAR</a>

</div>

</head>

<body>

<div class="left-container">

<h1 style="text-align:center;">Time Breaker 🕑</h1>

<div class="container">

<h2>Clock</h2>

<div id="clock" class="time-display"></div>

</div>

<div class="container">

<h2>Timer</h2>

<div id="timer" class="time-display">00:00</div>

<button onclick="startTimer()">Start</button>

<button onclick="pauseTimer()">Pause</button>

<button onclick="resetTimer()">Reset</button>

</div>

<div class="container">

<h2>Stopwatch</h2>

<div id="stopwatch" class="time-display">00:00:00</div>

<button onclick="startStopwatch()">Start</button>

<button onclick="pauseStopwatch()">Pause</button>

<button onclick="resetStopwatch()">Reset</button>

</div>

</div><br>

<div class="right-container">

<h1>Basic Warm-up Exercises</h1>

<div class="exercise-container">

<div class="exercise" onclick="openModal('Dumbbell Lateral Lunge', 'Works legs and glutes, improves balance.', '30 seconds each side', 'pexels-photo-999257.jpg')">

<img src="pexels-photo-999257.jpg" alt="Dumbbell Lateral Lunge">

<p>Dumbbell Lateral Lunge</p>

</div>

<div class="exercise" onclick="openModal('Seated Hamstring Stretch', 'Stretches the hamstrings and lower back.', '20-30 seconds each leg', 'pexels-photo-3771071.jpg')">

<img src="pexels-photo-3771071.jpg" alt="Seated Hamstring Stretch">

<p>Seated Hamstring Stretch</p>

</div>

<div class="exercise" onclick="openModal('Plank', 'Strengthens core and improves stability.', 'Hold for 30-60 seconds', 'images.jpg')">

<img src="images.jpg" alt="Plank">

<p>Plank</p>

</div>

<div class="exercise" onclick="openModal('Skipping', 'Improves coordination and cardiovascular endurance.', '1-2 minutes', 'Skipping-3-1200x900-1-1024x768.jpg')">

<img src="Skipping-3-1200x900-1-1024x768.jpg" alt="Skipping">

<p>Skipping</p>

</div>

<div class="exercise" onclick="openModal('Squat', 'Strengthens lower body muscles and improves flexibility.', '12-15 reps', 'pexels-photo-7187856.jpg')">

<img src="pexels-photo-7187856.jpg" alt="Squat">

<p>Squat</p>

</div>

<div class="exercise" onclick="openModal('Standing Quadriceps Stretch', 'Stretches the quadriceps and improves flexibility.', '20-30 seconds each leg', 'oi.jpg')">

<img src="oi.jpg" alt="Standing Quadriceps Stretch">

<p>Standing Quadriceps Stretch</p>

</div>

</div>

</div>

<div id="modal" class="modal" onclick="closeModal()">

<div class="modal-content">

<span class="close" onclick="closeModal()">&times;</span>

<h2 id="exercise-title"></h2>

<img id="exercise-img" src="" alt="Exercise" class="modal-content-img">

</div>

</div>

<script>

function openModal(title, imgSrc) {

document.getElementById("exercise-title").innerText = title;

document.getElementById("exercise-img").src = imgSrc;

document.getElementById("modal").style.display = "flex";

}

function closeModal() {

document.getElementById("modal").style.display = "none";

}

</script>

</body>

</div>

</div>

<script>

function updateClock() {

document.getElementById("clock").innerText = new Date().toLocaleTimeString();

}

setInterval(updateClock, 1000);

updateClock();

let timerInterval, timerTime = 0;

function startTimer() {

if (!timerInterval) {

timerInterval = setInterval(() => {

timerTime++;

document.getElementById("timer").innerText = new Date(timerTime \* 1000).toISOString().substr(14, 5);

}, 1000);

}

}

function pauseTimer() { clearInterval(timerInterval); timerInterval = null; }

function resetTimer() { pauseTimer(); timerTime = 0; document.getElementById("timer").innerText = "00:00"; }

// Stopwatch (Fast speed with step-by-step increments)

let stopwatchInterval;

let stopwatchTime = 0;

function startStopwatch() {

if (!stopwatchInterval) {

stopwatchInterval = setInterval(() => {

stopwatchTime++;

let date = new Date(stopwatchTime \* 1000);

document.getElementById("stopwatch").innerText = date.toISOString().substr(11, 8);

}, 200); // Updates every 200ms for a faster effect but step-by-step

}

}

function pauseStopwatch() { clearInterval(stopwatchInterval); stopwatchInterval = null; }

function resetStopwatch() { pauseStopwatch(); stopwatchTime = 0; document.getElementById("stopwatch").innerText = "00:00:00"; }

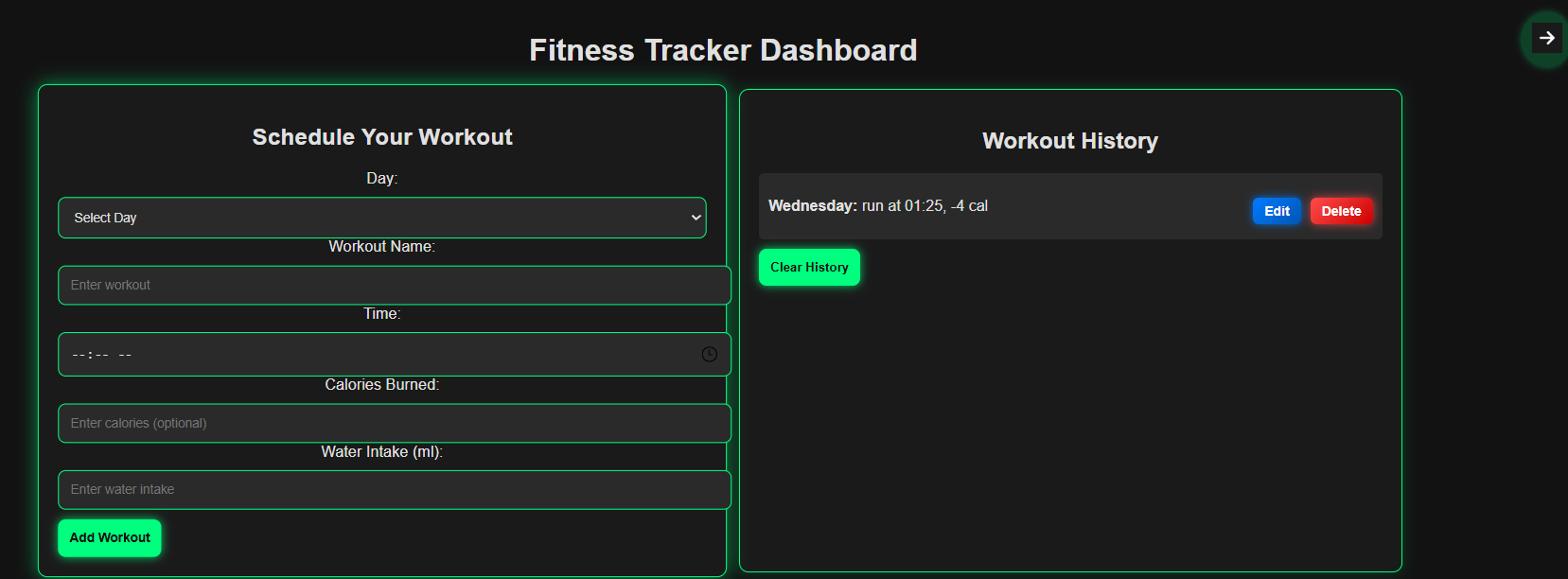
</script>

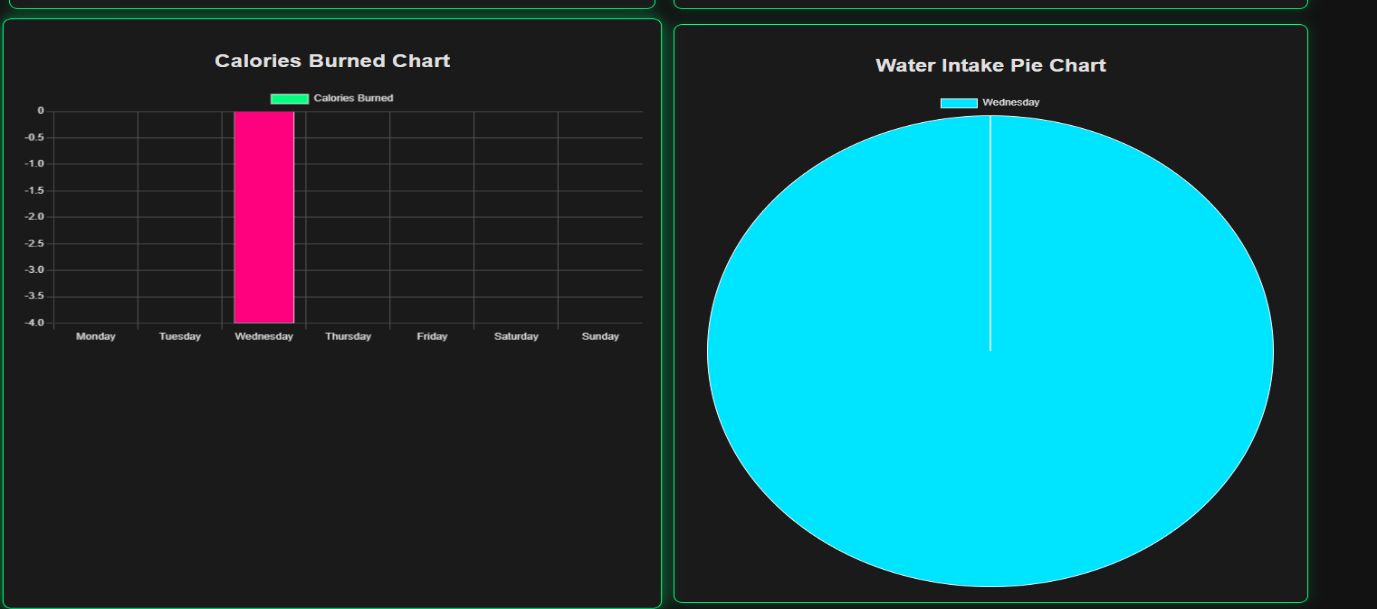
</body>

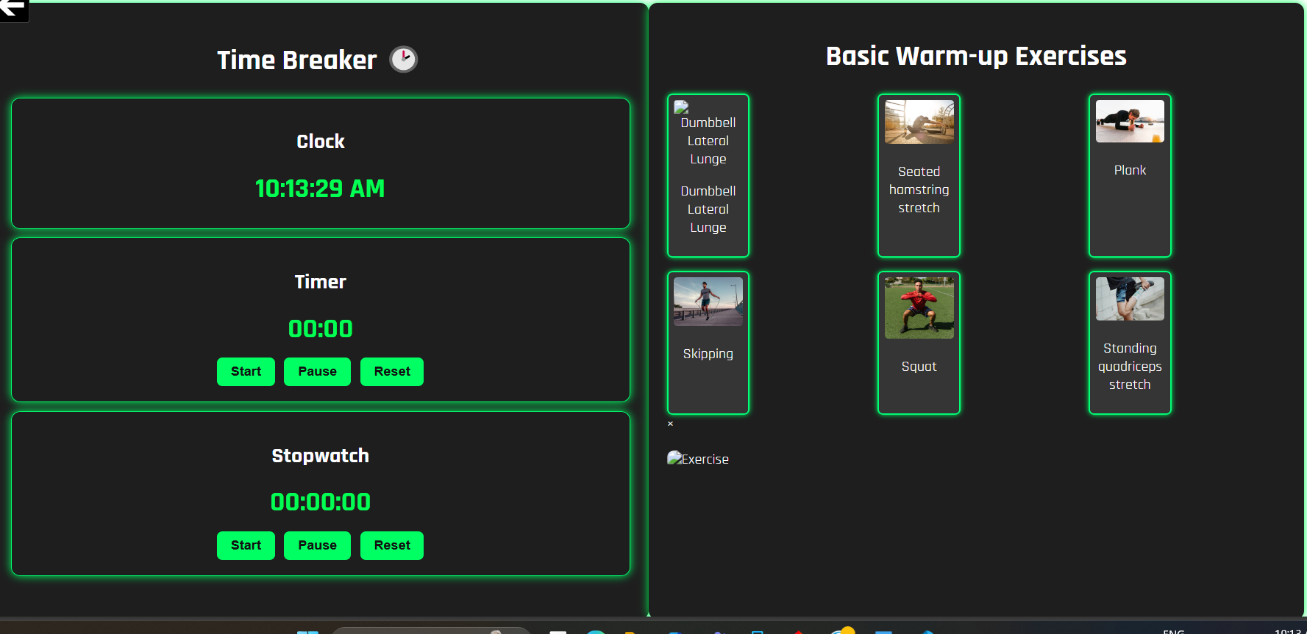
</html>

**SCREENSHOTS :**

**DASHBOARD :**

****

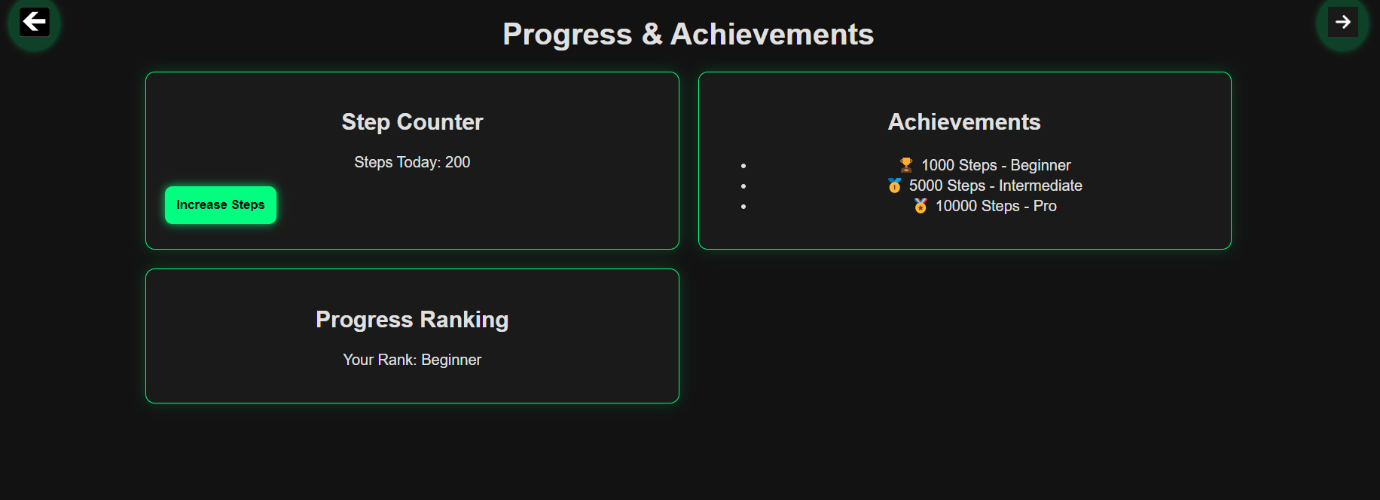
****

****

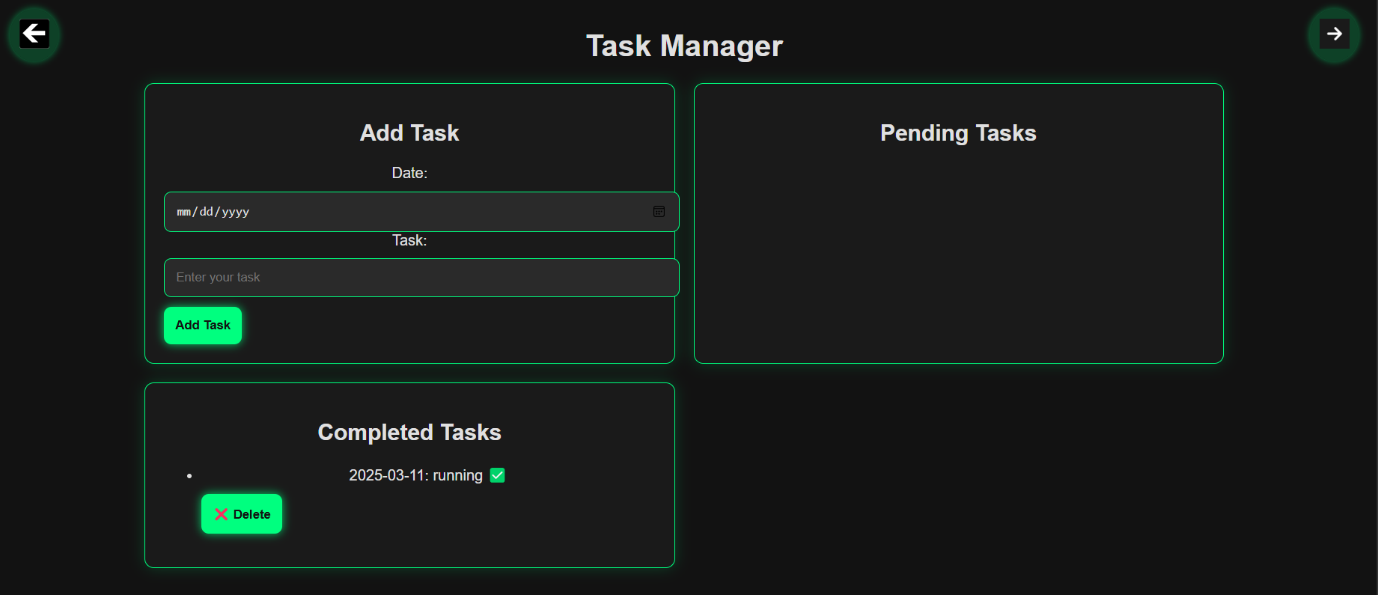
**HEALTH MONITOR:**

****

**PROGRESS & ACHIEVEMENTS:**

****

**TASK MANAGER:**

****

**Conclusion :**

The **Energy studioApplication** is a web-based system designed to help users monitor their workouts, water intake, and calorie burn efficiently. Developed using **HTML, CSS, and JavaScript**, it features an interactive **dark-themed UI with a neon effect**, a structured workout scheduler, real-time calorie tracking, and intuitive data visualization through charts.

The development process involved **system analysis, design, implementation, and rigorous testing** to ensure a **smooth user experience, high performance, and security**. The application is **responsive and compatible across different devices and browsers**, making energy studio accessible to users anytime, anywhere.

**Future enhancements :**

With **further enhancements**, such as **real-time cloud storage, AI-based fitness recommendations, and wearable device integration**, this system can be expanded into a **comprehensive personal health assistant**. Overall, the project successfully achieves its goal of **helping users stay fit and maintain a healthy lifestyle**. 1