**Hardware requirements**

* Iphone 6 or newer
* Desktop device
* Chrome browser
* Access to internet

**Script.js**

**Global Variables:**

* 1. timerInterval
  + Purpose: Stores the interval ID for the timer, allowing control over the timer intervals.
* 2. originalTime
  + Purpose: Holds the initial time value for the timer, serving as a reference for resetting the timer.
* 3. timeLeft
  + Purpose: Represents the current time left on the timer, updating as the timer counts down.
* 4. timerStarted
  + Purpose: Acts as a flag to track whether the timer is running or not.
* 5. timerExpired
  + Purpose: Serves as a flag to indicate whether the timer has expired.

**Functions:**

* 1. notifyMe()
  + Purpose: Displays a notification to the user when the timer expires or when permission is granted to display notifications.
* 2. updateTimer()
  + Purpose: Updates the timer display and handles actions when the timer expires, such as stopping the timer and triggering notifications.
* 3. startTimer()
  + Purpose: Initiates the timer, updating the timer display every second until the timer expires.
* 4. resetTimer()
  + Purpose: Resets the timer to its original state, stopping the timer interval and resetting time variables.
* 5. formatTime(seconds)
  + Purpose: Formats the time in minutes and seconds for display in MM:SS format.

**Event Listeners:**

* 1. DOMContentLoaded Event Listener
  + Purpose: Initializes the timer display and requests permission to display notifications when the DOM content is loaded.
* 2. Start/Reset Button Click Event Listener
  + Purpose: Controls the behavior of the start/reset button, either starting or resetting the timer based on its current state.
* 3. Event Listeners for Adjusting Timer Duration
  + Purpose: Allow users to adjust the timer duration by adding or subtracting minutes or seconds from the original time.
* 4. Event Listeners for Navigation Buttons
  + Purpose: Redirect users to different pages upon clicking on navigation buttons (e.g., logbook, routine, exercise library, documentation).

**Routine.js**

**Functions:**

* 1. toggleInputFields()
  + Purpose: Toggles the display of input fields between visible and hidden states.
* 2. removeEntry(index)
  + Purpose: Removes an entry from the stored data array in the localStorage based on the provided index.
* 3. addWorkout()
  + Purpose: Adds a new workout item to the workout list based on the input value entered by the user.
* 4. removeWorkoutItem(button)
  + Purpose: Removes a specific workout item from the workout list when the corresponding "Remove" button is clicked.
* 5. saveData()
  + Purpose: Saves entered data (day, type of workout, list of workouts) into the localStorage under the key 'routineData'.
* 6. displayEntry()
  + Purpose: Displays stored data entries from localStorage, generating HTML dynamically to represent each entry and appending it to the specified element.
* 7. goBack()
  + Purpose: Navigates the user back in the browser history when invoked.
* 8. window.onload
  + Purpose: Executes the displayEntry() function when the window is fully loaded, ensuring that stored data entries are displayed when the page loads.

**Event Listeners:**

No explicit event listeners are defined within this JavaScript file; however, functions like addWorkout() and removeEntry() are indirectly tied to DOM events (e.g., button clicks) through their respective HTML elements with inline event handlers.

**Logbook.js**

**Functions:**

* 1. toggleInputFields()
  + Purpose: Toggles the visibility of input fields with the class name 'input-fields' between visible and hidden states.
* 2. removeEntry(index)
  + Purpose: Deletes an entry from the stored logbook data array in the localStorage based on the provided index and updates the displayed logbook entries.
* 3. saveData()
  + Purpose: Saves entered data (workout name, sets, reps, weight, date) into the localStorage under the key 'logbookData', and clears input fields after saving.
* 4. displayEntry()
  + Purpose: Renders stored logbook data entries dynamically as HTML elements, displaying details such as workout name, sets, reps, weight, and date. Also, adds delete buttons for each entry to allow users to remove them.
* 5. goBack()
  + Purpose: Navigates the user back in the browser history when invoked.
* 6. window.onload
  + Purpose: Executes the displayEntry() function when the window is fully loaded, ensuring that stored logbook data entries are displayed when the page loads.

**Event Listeners:**

No explicit event listeners are defined within this JavaScript file; however, the functions removeEntry() and displayEntry() are indirectly tied to DOM events (e.g., button clicks) through their respective HTML elements with inline event handlers.