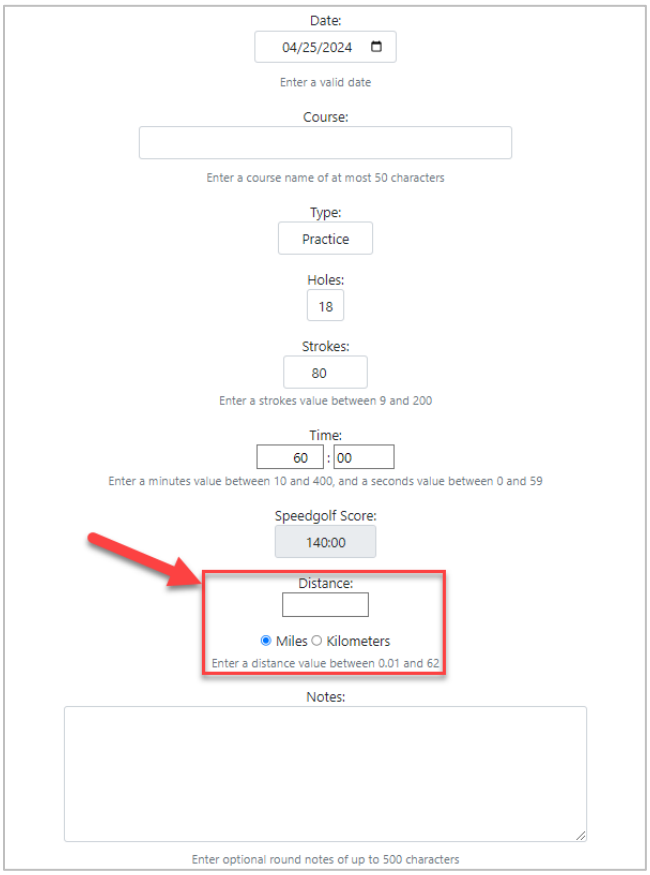
Implement Feature: Add "Distance" Field to Round

(Please take **up to 4 minutes** to read the following instructions aloud.)

Introduction

The following pages describe a series of three programming tasks that work toward the implementation of a new "Distance" field in the "Add Round" and "Edit Round" forms used to create and edit a round:



The "Distance" field allows users to enter the distance they ran during a speedgolf round, in either miles or kilometers.

To get started with this task, first check out the `addDistance` branch like so:

```
git checkout addDistance
```

Relevant Parts of the Code Base

In this task, you'll be working with three code files:

- `index.html`—This file contains the HTML code for the application. The form shown above is enclosed in a `<div>` element whose `id` is `roundsModeDialog`.

- `roundsMode.js`—This file contains the JavaScript code that defines the event handlers for Rounds Mode, including the event handlers for the form shown above. For example, the submit handler for the form shown above is called `GlobalLogRoundForm.addEventListener`.
- `main.js`—This file includes global variable declarations (prefaced by `Global`) for several key HTML elements defined in `index.html`. Functions and event handlers in `roundsMode.js` frequently use these global variables to refer to HTML elements in "Rounds" mode. Global variables for the form shown above are located under a comment block labeled "LOG ROUND DIALOG FORM."

Data Storage

SpeedScore does *not* interact with a back-end database. SpeedScore's user data are instead stored in the global variable `GlobalUserData` (declared in `main.js`) and in `localStorage`.

Prepare to Get Started

Before starting these tasks, please note the following:

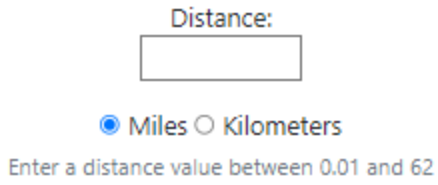
- You will complete the programming tasks **without the use of CoPilot**.
- You are free to **use the web to search for help or solutions**.
- Read all task instructions aloud *completely* before starting the task.
- Keep your focus on meeting the requirements of the current task. Do not work ahead.

Do you have any questions before you begin?

Task 1: Add Field, Radio Buttons and Label to Form

(Please take up to 2 minutes to read these task instructions aloud. Let us know when you are ready to begin the task.)

Add a distance field, two radio buttons, and label to the round form, as shown in this screenshot:



Solution Requirements

1. A "Distance field" must appear **centered** in the form below the "Speedgolf Score" field.
2. "Miles" and "Kilometers" radio buttons must appear **centered** below the Distance field.
3. Only one radio button may be selected at a time: Clicking on the "Kilometers" radio button selects the "Kilometers" radio button and unselects the "Miles" radio button, and vice versa.
4. A label "Enter a distance value between 0.01 and 62" must appear centered below the radio buttons.
5. The distance field must be empty or display only numbers between 0.01 and 62, in increments of 0.01.
6. By default, the "Miles" radio button is selected.

Testing Your Solution

You may determine the correctness of your solution at any time by running a test suite like so:

```
npm run test task1
```

To practice this, run the test suite **right now**.

Completing the Task

You are done with the task when all tests in the test suite pass.

Please let us know when you are ready to begin the task.

Please await further instruction before proceeding to the next page.

Committing and Pushing Your Solution

Commit and push your code solution for Task 1 like so:

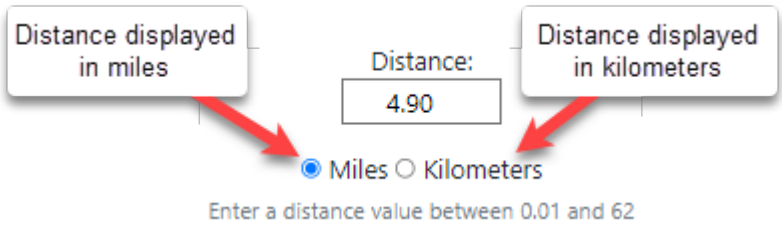
```
git add .
git commit -m "Task 1"
git push origin
```

Please await further instruction before proceeding to the next page.

Task 2: Convert Distances between Miles & Kilometers

(Please take up to 2 minutes to read these task instructions aloud. Let us know when you are ready to begin the task.)

When the user clicks on the "Miles" or "Kilometers" radio button, the value displayed in the "Distance" field should toggle between miles and kilometers, as shown in this screenshot:



Solution Requirements

1. The "Distance" field must accept only numeric values between 0.01 and 62.
2. Numeric values entered in the Distance field must be rounded to **exactly** two decimal places.
3. When the "Kilometers" radio button is selected, the value in the "Distance" field is converted to kilometers.
4. When the "Miles" radio button is selected, the value displayed in the "Distance" field is converted back to miles.

Testing Your Solution

You may determine the correctness of your solution at any time by running a test suite like so:

```
npm run test task2
```

To practice this, run the test suite **right now**.

Completing the Task

You are done with the task when all tests in the test suite pass.

Please let us know when you are ready to begin the task.

Please await further instruction before proceeding to the next page.

Committing and Pushing Your Solution

Commit and push your code solution for Task 2 like so:

```
git add .
git commit -m "Task 2"
git push origin
```

Please await further instruction before proceeding to the next page.

Task 3: Save "Distance" field with Round

(Please take up to 2 minutes to read these task instructions aloud. Let us know when you are ready to begin the task.)

When the "Add Round" or "Update Round" button is clicked, the distance field should be saved with the round data to the user data object and local storage.

Solution Requirements

1. When the user clicks on the "Add Round" or "Update Round" button, save the "Distance" field **in feet** to a property called `roundDistance` in the user's object in Local Storage.
2. When the user chooses to view/update an existing round, load the value of the corresponding round's `roundDistance` property from Local Storage into the "Distance" form field.
3. Display the value in *miles* by default, with the "Miles" radio button selected.

Testing Your Solution

You may determine the correctness of your solution at any time by running a test suite like so:

```
npm run test task3
```

To practice this, run the test suite **right now**.

Completing the Task

You are done with the task when all tests in the test suite pass.

Please let us know when you are ready to begin the task.

Please await further instruction before proceeding to the next page.

Committing and Pushing Your Solution

Commit and push your code solution for Task 3 like so:

```
git add .
git commit -m "Task 3"
git push origin
```

Congratulations! You are done with all tasks for this feature!