Sprint 3 Scoping
Team Four Real

- **Team Members**
  Yu-Hsuan Lin, Yi-Jie Chou, Luojia Zhao, Chienchia Chiu

- **GitHub Repository**
  https://github.com/zhaoluojia/cs5500_group_project

- **Project Board**
  https://github.com/users/zhaoluojia/projects/1/views/3

- **Project Name**
  Exercise Manager

- **Sprint 3 Scope**
  In Sprint 2, we'll focus on creating the UI, deploying the app to the cloud, and setting up a CI pipeline to ensure code quality and correctness.

- **Sprint 3 Tasks**

1. **Create UI**
   We'll use the React library to develop three different web pages that will display data showcasing three different user stories. These pages will be built using the REST API that we designed and implemented in previous sprints. Wireframes and specifications for each page have already been created as below.

   User story1:
   As an exercise lover, I want to get the calories burned in my activities by inputting the duration/type (walking/jogging/cycling/kayaking) of exercise and weight.

   - Webpage description:

   A page asking for input of the exercise info and output the calories and a success message:

   - Input area:
     Username(String) / Date(yyyy-mm-dd) / Duration(hh:mm) / Exercise Type.
   - Output area:
     If input valid: Showing the calories burned with message "exercise added successfully. You burned xxx calories in this exercise!".
     If input invalid: Showing empty/wrong next to the input column is contains invalid (empty) input.

   - REST API to be used:

   ```
   public ResponseEntity<String> createExercise(@PathVariable Long userId,
       @RequestParam String exerciseName,
       @RequestParam("date") @DateTimeFormat(pattern = "yyyy-MM-dd") Date date,
       @RequestParam Double duration)
   ```

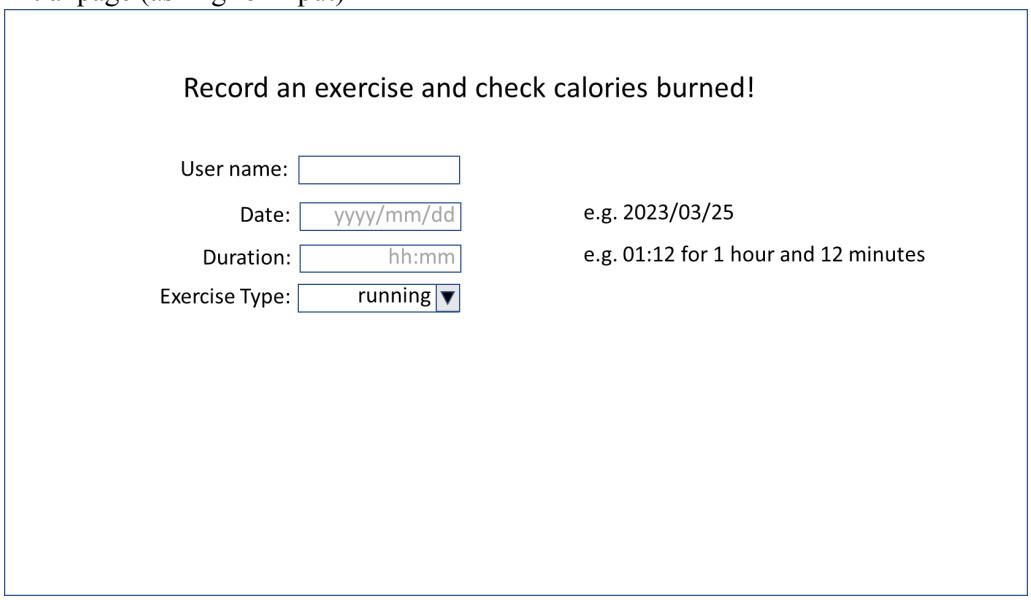
   The following two REST APIs should be created:

   ```
   public Double getCaloriesOfExercise (
       @RequestParam String exerciseName,
   ```

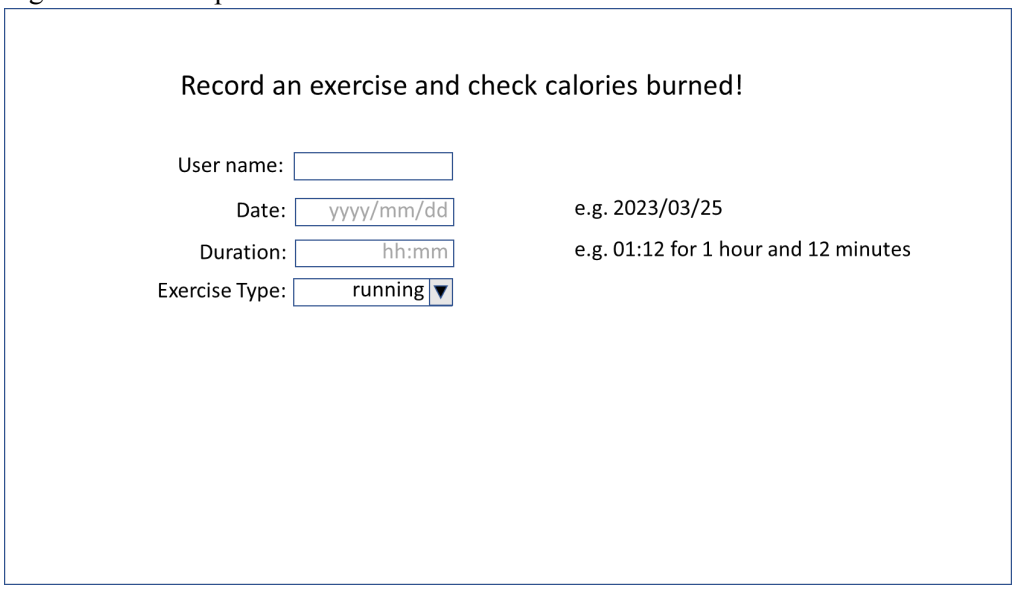
```
    @RequestParam Double duration) {
                return userService.calculateCalories(weight, exerciseName, duration);
        }

 public User getUserByUsername(
    @RequestParam("username") String username) {
  return userService.getUserByCredentials(username);
        // this corresponding service ".getUserByCredentials(username)" is to be created too.
 }
```

■ Wireframes

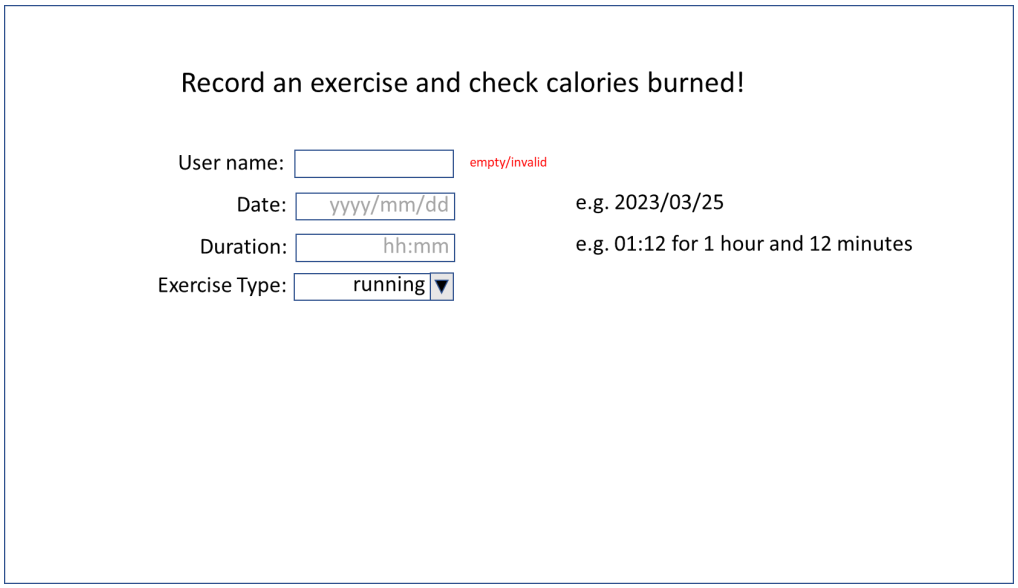
Initial page (asking for input)



Page for a valid input



Page for invalid/empty inputs



User Story2:

As a person on a diet, I want to set a goal and compare the actual calories burned/actual duration run between a certain period.

- Webpage description:

A page asking for input of the goal info and output the progress with message:

Step 1 (Part A):

Asking the user to input the username, if valid, Part B would show, if not, there would show an error message “empty/invalid”.

Step 2 (Part B):

Asking the user to select a goal, which is the calories goal or duration goal. After selecting the button, Part C would show.

Step 3 (Part C):

Asking the user to input the goal, if the user selected the calories goal earlier, the user could input the calories goal; if the user selected the duration goal earlier, the user could input the duration goal. If the input valid, Part D would show, if not, there would show an error message “empty/invalid”.

Step 4 (Part D):

Final page: Showing the total calories burn / total duration, a progress bar, and a message that shows how many percentages the user completed.

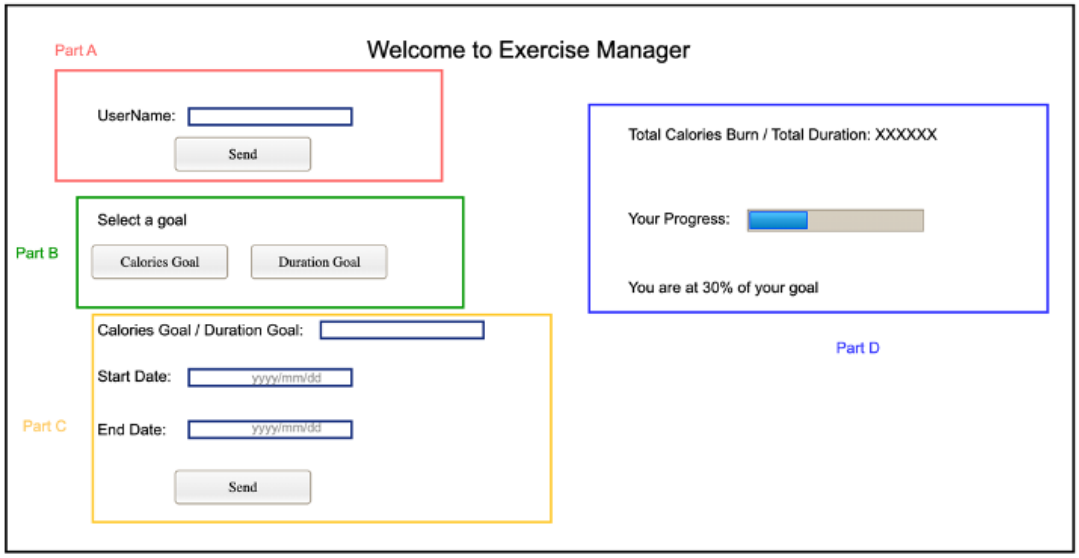
- REST API to be used:

public Map<Date, Double> getDurationTotalBetweenDates(@PathVariable Long userId, @RequestParam("startDate") @DateTimeFormat(pattern = "yyyy-MM-dd") Date startDate, @RequestParam("endDate") @DateTimeFormat(pattern = "yyyy-MM-dd") Date endDate)

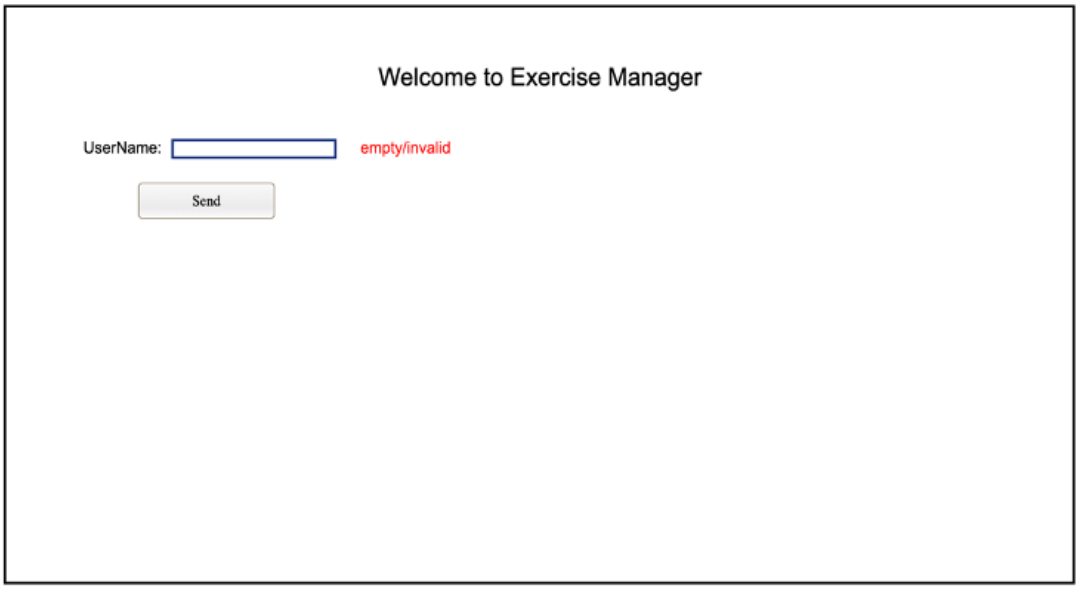
public Map<Date, Double> getCaloriesTotalBetweenDates(@PathVariable Long userId, @RequestParam("startDate") @DateTimeFormat(pattern = "yyyy-MM-dd") Date startDate,

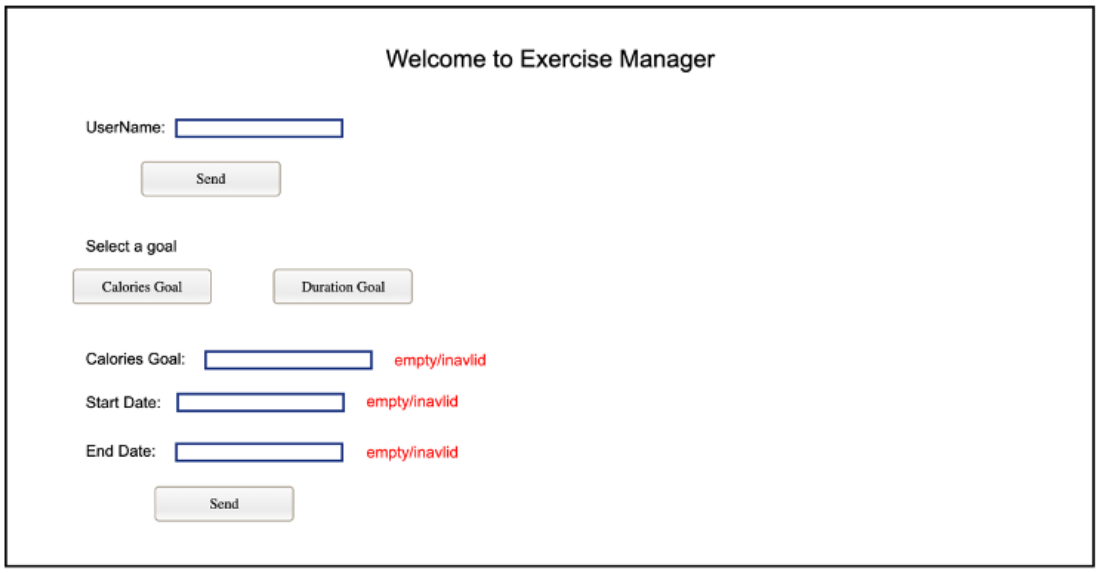
@RequestParam("endDate") @DateTimeFormat(pattern = "yyyy-MM-dd") Date endDate)

- Wireframes:

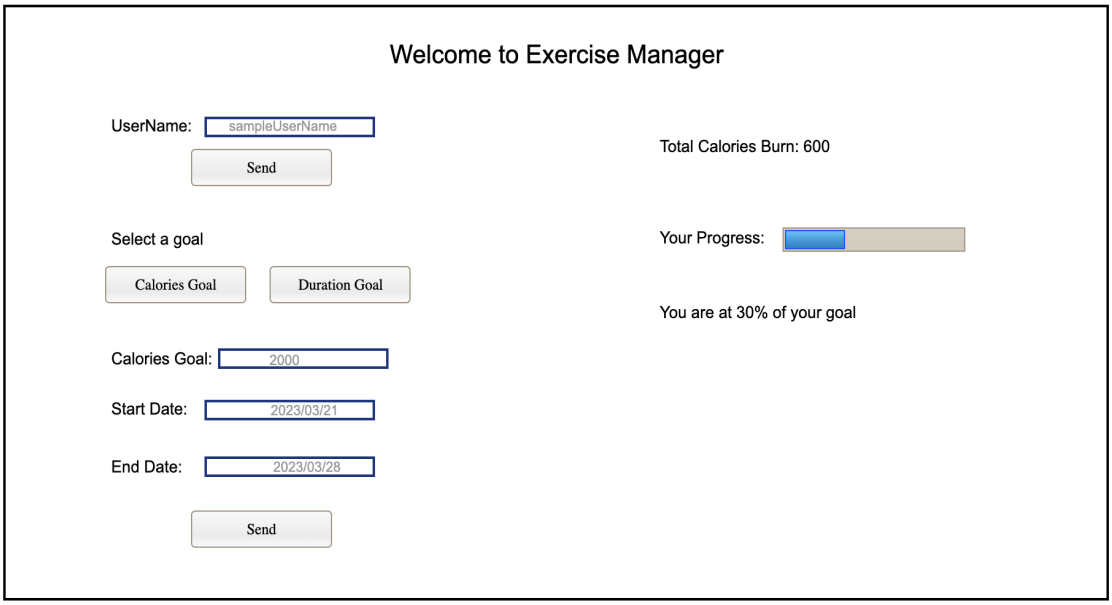


Samples of invalid input:





Sample of final page:



User Story3:

As an exercise lover, I want to get a weekly report about the past 7 days' exercise activities , with charts showing me each day's total exercise duration /calories , and advice on which day I should exercise more.

Analysis of the smallest calories/exercise duration between two dates is implemented in UserService.

Planned to use API like Spring MVC to visualize each day's duration/ calories by charts when creating frontend.

■ Webpage description:

A page showing charts with weekly summary of calories and duration, and suggestions om which days should exercise more:

- Charts: Showing a daily calories burn chart and a daily duration chart
- Summary: Showing suggestions

■ REST API to be used:

public Double getDurationTotalBetweenDates(@PathVariable Long userId, @RequestParam("startDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date startDate, @RequestParam("endDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date endDate)

public Double getCaloriesTotalBetweenDates(@PathVariable Long userId, @RequestParam("startDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date startDate, @RequestParam("endDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date endDate)

public Double getSmallestDurationBetweenDates(@PathVariable Long userId, @RequestParam("startDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date startDate, @RequestParam("endDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date endDate)

public Double getSmallestCaloriesBetweenDates(@PathVariable Long userId, @RequestParam("startDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date startDate, @RequestParam("endDate") @DateTimeFormat(iso = DateTimeFormat.ISO.DATE) Date endDate)

■ Wireframs

2. **Deploy the app to the cloud**
   Our database has already been deployed using MongoDB Atlas, and we'll now deploy our frontend and backend to AWS EC2.

3. **Setup CI pipeline**
   To ensure code quality and correctness on every push to main, we'll set up a Continuous Integration pipeline using Github Actions.