Fitness – a Web App with Next.js

Introduction

A new customer who runs a fitness center where several personal trainers work with their individual clients has contacted us. The owner of the fitness center wants us to develop a web app to be used by the personal trainers to create individual workout programs for their clients. Another team does the backend development, and your team can use their work when you develop the front end. The backend team have had some problems, but their first release is now ready for the initial development of the front end.

Backend api

The backend is documented with swagger and the api documentation can be found at https://assignment2.swafe.dk/swagger/index.html

Notice: you do not need to use all the api endpoints!

If you discover any errors in the backend, you can report them to per@ece.au.dk or jba@ece.au.dk

Technology requirements

The web app must use **Next.js** and must include both server-side and client-side components. The app must be deployed to a public cloud (such as Vercel).

Functional requirements

Develop the front-end for a fitness web app. Personal trainers should be able to create workout programs like the one shown beneath. A workout program is a collection of exercises (workouts) that each have a name, description, number of sets and number of repetitions or time.

Basic functionality:

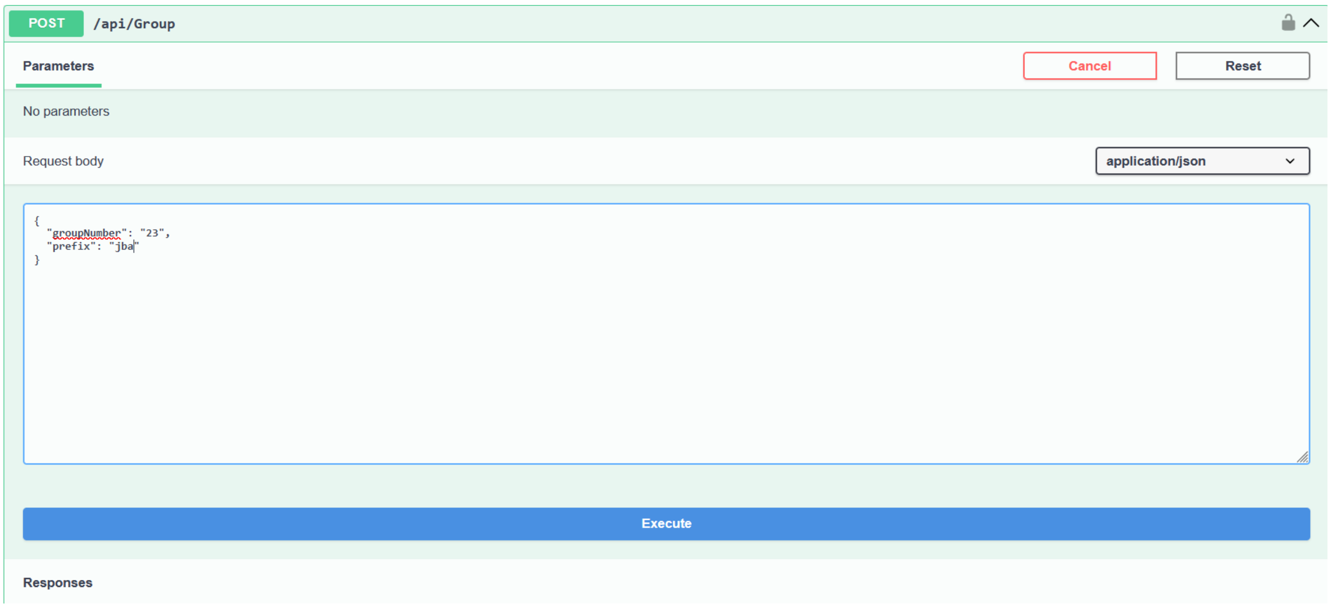
- The user can login
- The manager
    - Can create users (personal trainers)
- A personal trainer
    - Can create users (clients)
    - Create a new workout program for a client
    - Can add new exercises to a workout program
        - An exercise has a name, a description, number of sets and number of repetitions or the time it should last.
    - See a list of workout programs.
    - See a specific workout program.
    - See a list of clients.
- A client
    - Can see his/her workout program.
    - If the user has more than one program then the app will show a list of programs, and the user can then select the program to be displayed.

Workout program example.

Exercise	Description	Set	Reps/time
Squat	Stand with your feet spread shoulder-width apart. Lower your body as far as you can by pushing your hips back and bending your knees. Pause, and then slowly push yourself back to the starting position.	3	20
Push ups	Place your hands on the floor with legs straight out behind you resting on your toes. Bend your arms and slowly …	3	10
Plank	Place your elbows on the floor shoulder-width apart with legs stretched out behind you so only your elbows and toes are in contact with the ground. Use your abdominal muscles to keep …	1	30 sec

Setup before you begin

Go to the swagger url https://assignment2.swafe.dk/swagger/index.html and make a post with your group info in the group endpoint. Group number should be your group number and the prefix can be whatever you want, but don't make it too long – it will be prefixed to the users created.



```
{
  "groupNumber": "23",
  "prefix": "jba"
}
```

In this example the Manager email will be jba_boss@fitness.dk

These users are seeded into the database

```
                // Manager
                    Email = "{prefix}_boss@fitness.dk",
                    FirstName = "Manager",
                    LastName = "The Boss",
                    AccountType = Manager,
                    Password = "asdfQWER",

                // Personal trainers
                    Email = "{prefix}_m@fit.dk",
                    FirstName = "Superman",
                    LastName = "Mars",
                    AccountType = PersonalTrainer,
                    Password "aQ",

                    Email = "{prefix}_w@fit.dk ",
                    FirstName = "Superwoman",
                    LastName = "Venus",
                    AccountType = PersonalTrainer,
                    Password = "aZ"

                // Clients
                    Email = "{prefix}_c1@fit.dk ",
                    FirstName = "John",
                    LastName = "Doe",
                    AccountType = Models.Enums.Role.Client,
                    Password = "aA",
                    PersonalTrainerId = 2

                    Email = "{prefix}_c2@fit.dk ",
                    FirstName = "Jane",
                    LastName = "Doe",
                    AccountType = Models.Enums.Role.Client,
                    Password = "aA",
                    PersonalTrainerId = 3
```

Formalia

- Group size: 1-4 people
- Deadline: november 28, 2025

Submission

Your submission consists of 3 parts:

1. A video where you demonstrate that your app fulfills the requirements.
2. Your solution as a zip package as described below.
3. Link to where your app is deployed.
4. Your chosen prefix.

Before submitting your solution, do the following:

1. Delete the `node_modules` folder in the workspace root folder
2. Add a file `participants.txt` and insert a new line for each participant with the AUID and name of each member separated by whitespace
3. Add `participants.txt` to the root folder of your application
4. Archive and compress you application using `zip`. All other formats will result in a request for resubmission.
5. The filename should be named `<AUID_PARTICIPANT_1>-<AUID_PARTICIPANT_2>-<AUID_PARTICIPANT_3>.<ARCHIVE_COMPRESS_FORMAT>` *Example: Alice with AUID* `au01248` *and Bob with AUID* `au84210` *creates a compressed archive named* `au01248-au84210.zip` *and uploads it to Brightspace*

Example `participants.txt` contents:

```
au01248 Alice
au84210 Bob
```