APSIS FRONTEND coding assignment  
Description  
The use-case is a company-wide steps leaderboard application for teams of employees: picture that all employees are grouped into teams and issued step counters. This application needs to receive and store step count increments for each team, and calculate sums.  
  
Focus on showing craftsmanship and code quality over full functionality.

User Stories  
Imagine an API that exposes methods that supports following User Stories:

|  |  |
| --- | --- |
| 1 | As a User I want to be able to create a new counter So that steps can be accumulated for a team of one or multiple employees |
| 2 | As a User I want to be able to increment the value of a stored counter So that I can get steps counted towards my team's score |
| 3 | As a User I want to get the current total steps taken by a team So that I can see how much that team have walked in total |
| 4 | As a User I want to list all teams and see their step counts So that I can compare my team with the others |
| 5 | As a User I want to list all counters in a team So that I can see how much each team member have walked |
| 6 | As a User I want to be able to add/delete teams  So that I can manage teams |
| 7 | As a User I want to be able to add/delete counters  So that I can manage team member's counters |

Backend  
Imagine that there exists a REST API supporting the above user stories.

## Remarks

* Make the API nice to use for a hypothetical client developer.
* Counter states only have to be stored during the application lifetime and can be  
  forgotten on shutdown (you don’t need to implement persistent storage layer).
* Bonus points:
* nice README and Open API/Swagger specifications
* if you can host a running app somewhere (Heroku/AWS/Azure) where we  
  can play around with it
* The API should follow the RESTfull API principles (<https://restfulapi.net/>).

Frontend  
Create a web application using the API backend.  
  
Remarks

* Angular8 or ReactJS, both with Typescript
* Use a CSS preprocessor library of choice (SCSS)
* Unit testing with Jest and Jasmine or similar would be awesome
* Bonus points: use Redux state management
* Bonus points: use Reactive programming with RxJS

# Additional questions

# You do not need to actually implement support for the below items, just have an idea for how the app would be changed to support each one. We will discuss them during a subsequent code review session.

## Authentication

* How would you ensure that only authorised users can submit and/or retrieve  
  data?
* How would you then add support to allow different users to only update specific  
  counters? Or perform only specific operations?

# Last words and Bonus points

* Focus your time on areas you are the strongest in.
* Keep the code clean with a structure that is object oriented and easy to read and  
  to test.
* Feel free to use any package manager that fits you (nuget, npm, yarn, go...).
* Make sure your code builds and runs on any developer’s computer.
* Bonus points for a good README.md (something you should want to read)

**Good luck and have fun!**