Technical Test - Front-End Developer

# Objectives and Guidance

Below is a programming problem that we’d like you to solve.

For your solution, you should:

* Use any external libraries you deem necessary, as long as you include the necessary tooling and instructions to make installing them easy peasy.
  + This includes any frameworks that you deem appropriate
  + You can assume the presence of standard package managers (e.g. npm); if you’re using something more exotic, include links for how to install on a Mac and PC (both).
* Write your solution in either JavaScript or TypeScript - again, noting that your app needs to build and run out of the box with minimal effort.
* Include automated tests, especially unit tests, as and where you deem them appropriate, given the problem statement below.
* Scaffold the application in a way that anticipates it growing into something larger… without too much future proofing.
* Assume that it only needs to run on the latest version of Chrome.
* Ensure it runs nicely in a development environment, but skip packaging it up for deployment to production (e.g. don’t spend any extra effort on minification, obfuscation, etc.).
* Include a README, which includes a brief explanation of your design and assumptions, as well as instructions on how to run your application (including how to run the tests).

We assess a number of things, including the design of your solution, program correctness, coding style, user consideration, and general maintainability. While it is a small-ish problem, we expect you to submit what you believe is production-quality code – code that you’d be able to run, maintain, and evolve. You don’t need to “gold plate” your solution; however we are looking for something more than a quick and dirty prototype. You should not spend more than 3 hours total on this exercise - as you plan your work, if you are considering how to trade off between scope and quality, *you should err on the side of quality*. For any scope that you have cut, please list it in your README, along with a high-level strategy for how you would continue the work.

As a general rule, we allow three days from the date that you receive these instructions to submit your code, though you may request more time if needed. If you have any questions about the code, as it relates to your interview process, please contact us.

Good Luck!

# 

# Problem: Field Support for Batteries

From time to time, our in-country tech teams are called upon to perform school visits to address issues that they find. We collect a bunch of data on an on-going basis about these devices and we’re interested in building a tool to identify teacher tablet batteries that are in need of replacement.

For this problem:

* We’re providing a JSON data file that represents a week’s worth of battery data for a small number of schools. Integrate with this file like you would against any API, including any relevant service layers - just don’t worry about any of the network or security aspects that a real API would entail. Also, the data size for the GET, if you were to pull all the data for all schools would be huge - don’t worry about that either.
  + The employeeID is a unique identifier for each employee. It tells you who was logged into the device when the relevant measurement was taken - but doesn’t affect the battery calculation. It’s possible that different users are using the same device.
  + serialNumber represents a unique device and the basis of the calculation
* A battery is considered to be in need of replacement if it uses more than 30% of its battery per day (on average). These are e-ink devices that are expected to last a week or more between charges.
  + calculations should be made between recorded data points, irrespective of the day.
    - there is no reason to believe that data will be recorded at the same time on two separate days. power levels are only recorded when the device touches the network.
    - if there were only two readings for a given device, one at 9 AM reading 100% and one at 9 PM on the same day reading 90%, then the average daily battery usage would be 20% (10% over 12 hours == 20% over 24 hours).
    - If there were a third reading at 9 PM for the *following* day at 80% (and no other readings), then the daily average for that device would be 13.3% (20% over 36 hours).
  + if the battery level **increases** between measurements, assume that the device was charged between readings and the change should be excluded from the calculation.
    - If there were a fourth reading at 10 PM at 100%, the daily average for that device would remain at 13.3%.

The task:

* Write a web application that identifies the **schools** with the highest number of battery issues, so that the field team knows which schools to visit first
* For each school, identify which devices are unhealthy and in need of battery replacement

Note: If this were an actual project, there would be a UX designer/product designer working *with* you to flesh out a great user experience… but for this exercise, we’re asking you to come up with your own design for the screen(s), based on the requirements above. Don’t worry about making it pixel perfect - just make sure it’s something reasonable for the stated users.