Volta Charging
Software Engineer Code Challenge Rubric August 3, 2016

Area	Points			
	0	1	2	3
Client (HTML/MVC)	No MVC used. Client crashes or is unusable. No data displayed and no functionality.	Candidate used any MVC; may have missed many best practices. Only data displays on screen.	Candidate used any MVC; may have missed a best practice. Data displays on screen and filter works.	Candidate implemented React or Angular solution using best practices. Data displays on screen and filter sorts by location name.
Styling (CSS)	There is no CSS included in the project or design is completely unacceptable.	Candidate made an attempt at following the example, but clearly did not succeed. CSS is not DRY or reusable.	Candidate did not use flex-box. Candidate matches >75% but less than 90% of provided example. CSS is generally reusable.	Candidate used flex-box to complete design. Matches >90% of provided example or improved upon design. CSS is reusable and DRY.
Server	Server does not run or no server.	Non-JavaScript server or very little code was written because a generator was used. Endpoint serves data.	Simple server uses JavaScript. Has one GET endpoint for client and reads data from provided database.json file.	Simple server uses JavaScript. Has one GET endpoint for client and reads data from provided database.json file. Serves static assets (client) to localhost.
Documentation	No RUNME or RUNME does not include info on how to run locally.	RUNME is barebones. It skips many steps for running locally. May have some writing errors. No use of Markdown.	RUNME communicates how to run the project, but may lack some styling or skip a few steps. Language is clear and few writing errors.	RUNME uses Markdown to clearly communicate how to run the project locally. Writing is without errors.
Version Control - Frequency (>5) - Capture major milestones - Well articulated commit messages	Did not use Git or follow GitHub forking steps. Project only has one commit for all code.	Candidate correctly forked project and used Git. Commits met 1 of 3 criteria.	Candidate correctly forked project and used Git. Commits met 2 of 3 criteria.	Candidate correctly forked the project per instructions and used Git for version control. Commits met 3 of 3 criteria.
Architecture	All code in one file or no file structure for many files.	MVC components were used, but not leveraged to their full potential. There is only a basic folder structure (client/server). Variables are named poorly.	MVC was leveraged appropriately for the project. There is some folder structure, but it is not scalable. Code is generally DRY and variables have thoughtful names.	Code was broken into reusable MVC components and files. Files were organized into a logical folder structure. Code was DRY. Functions and variables have thoughtful names.

Applicant Name:_____ Reviewer: _____

Submission Date: _____ Review Date: _____

Score: / 18

Suggest Pass: Y / N