**Job description**

We are looking for a versatile technical addition to our development team here in Seattle. The position offers an extensive amount of ownership and influence to our development processes as we scale the largest team based fundraising solution in the country. You will be called upon to not only write the code, but also be a key driver in our product's development.

**Responsibilities**

**Our Stack Overflow Company Page**

* Implement advanced front end features for our primary product using React, EmberJS, and Ruby on Rails
* Interpret requirements of implementation with a versatile group of product designers and engineers
* Ability to digest feature specifications and user stories into technical implementation
* Collaborate with cross-functional teams to define, design, and release new features
* Feature level testing verifying code for robustness, including edge cases, usability, and general reliability
* Address and prioritize production level bugs and application performance improvements
* Continuously evaluate and implement new technologies to maximize development efficiency
* Be comfortable releasing production ready code on tight deadlines with high visibility
* Communication above all, do you know when to speak up and ask for help or feedback?

**Requirements**

* Formal software education in front end development and design
* 1+ years of professional software development with extensive experience in the front end
* 6+ months of experience using React in a medium-to-large production-grade web application

**Nice-to-Haves**

* In-depth knowledge of React, including experience with scalability issues and system architecture
* Experience with EmberJS, AngularJS, BackboneJS, or any other similar modern Javascript MVC framework
* Experience with Sketch for consolidating and implementing product designs, Photoshop optional
* Familiarity with Ruby on Rails or any modern MVC framework
* Experience building an API-driven Javascript application
* An understanding of development in test driven environments
* Ability to write Javascript tests with Mocha, Jasmine, Tape, or any other related Javascript test framework
* Ability to write unit tests and function feature tests with Capybara / Selenium
* Understanding of Git for source code management and revision control
* An adaptable attitude and understanding of the Agile software development methodology
* Ability to reevaluate priorities on a tight timeline
* Self-motivated with a team-first mentality
* Ability to work at a results-oriented, product-driven startup

**Our Ideology as we Build, Create, and Innovate**

* Our workflow is cut, dry, and efficient. Adaptation is key, can you join this process?
* We only work on the most important things at any given point in time
* We then break those things into the smaller manageable tasks for individuals to work on autonomously
* Individual's strong suits are considered in assigning tasks and accomplishing the collective team goal
* Every morning the team recaps progress and blocking points for about 10-20 minutes, directed by the team lead
* Communication is the upmost of importance, are you comfortable asking for help, and open to accepting constructive feedback?
* Formally and collectively understand once a MVP sprint has been completed, to reevaluate the next set of priorities. Sometimes as frequently as weekly.

**Why This Opportunity is Unique**

* Be a core developer and architect on the largest team based fundraising solution in the country
* Define your process and foster ownership over our team's technical success
* Work on software that gets deployed to a massive audience on a large scale
* Contribute to a software solution that makes a lasting impact in the communities we work with, year after year.

**We Offer**

* Competitive Salary
* Health Benefits Included
* Free Sports League Entrance: Yes we have company kickball & dodgeball teams!
* Orca Public Transit Card
* Quiet, dedicated, awesome work space environment with an amazing view of Lake Union.
* Relocation Assistance is also offered for the successful candidate.

1. The most influential book I’ve ever read on web or any kind of development has been The Pragmatic Programmer. It’s a book of useful wisdom and practical advice on how to go about actually thinking of development, and to think of the process of work itself as a problem worth consideration.
2. The most ambitious program I have built is an app that helps you find restrooms in your area. I built it as an exercise in google maps APIs and it quickly grew beyond that. I realized that the app itself was extremely handy, and too much of a good idea not to expand upon. I’m still developing it now. At Snap! Raise you place a great deal of importance on the ability to take ownership of a project and to make it your own. This is my how I feel about projects that I am developing on. I feel like anything I work on should be excellent.

The main issue I faced was getting google maps API to fit into the Knockout framework in a way that is logical and makes for readable code. This was especially difficult because coming into the project I was a novice in Knockout and found that many of the deeper features of the framework are not well documented. To overcome this I ended up reading a lot of code written by knockout and google maps experts. Not many projects have been written in Knockout which used google maps API, so I had to infer many things from what I was reading. This required me to really stretch my skills as a problem solver.

Knockout is a great tool because the view model paradigm makes a lot of sense, and it does a great job of implementing declarative bindings and dependency tracking. React, for instance, does not have two way databinding. In the future I will probably choose to develop in something else though. The reason is that I prefer that my framework be DOM manipulation library agnostic and for it to support things like view animations and server side rendering.

1. def MakeList(list\_of\_strings):

answer\_array = []

answer = ''

for string in list\_of\_strings:

answer\_array.append('<ul>')

answer\_array.append(string)

answer\_array.append('/ul>')

answer\_array.append(' ')

answer = answer.join(answer\_array)

return answer

The reason I’ve built this function using array appends and joins is because I wanted to limit the number of objects held in memory. Using a concatenation function would generate a new string object each time an assignment occurs. While this is fine for a smaller operation, supposing we had millions of lines of items in our array “list\_of\_string” this would drastically and needlessly increase the time of said operation. I enjoy iterating over a solution and fine tuning it to be better because continually improving a product is inherently an exciting process for me.

If the string data was provided by user input we would need to pass it through a good sanitization library, like bleach. I would implement it like this:

Import bleach

def MakeList(list\_of\_strings):

answer\_array = []

answer = ''

for string in list\_of\_strings:

answer\_array.append('<ul>')

answer\_array.append(bleach.clean(string))

answer\_array.append('/ul>')

answer\_array.append(' ')

answer = answer.join(answer\_array)

return answer

1. I think that among the most dangerous attacks are SQL injection and, cross site scripting. Two of the biggest problems with web based applications are the number of interacting pieces as well as the large attack surface. Web applications work through many layers, converting data for different systems and typically will have many endpoints and features open to user manipulation. What this means is that opportunities for injecting crafted data are numerous. A classic hack involves poorly implemented SQL databases. Should a web application improperly sanitize user submitted data they could easily drop tables or gain access to unauthorized information. I think best practices are to clean data on both the client and server end, and to implement white listing when possible. Cross site scripting occurs when user submitted code is not properly sanitized. This allows the user to access, manipulate, and transmit data between different permissions. A good example is a web forum that allows users to submit html code that is then executed. A malicious user could add code such as script tags that load a JavaScript file that steals user information.
2. from flask import Flask

app = Flask(\_\_name\_\_)

import json

import random

@app.route('/')

def hello\_world():

return 'Hello World!'

@app.route(‘/dice/’)

def roll\_dice():

return json.dumps({'dice 1': random.randint(1,6), 'dice 2':

random.randint(1,6)})

if \_\_name\_\_ == '\_\_main\_\_':

app.debug = True

app.run()

1. I think that constant improvement and education are critical to developing within a career. A year from now I intend to have mastered a much broader body of knowledge including more frameworks and programming languages. For example, I want to be able to do complex work in Java and C++. Mastering challenging new technologies and skills are basically the main reasons I enjoy programming. To me this struggle and the satisfaction of surmounting it are my greatest joys.