Hi,

Thanks for taking the time to complete our code challenge! Below you will find 2 exercises. Please enter your answers into the Greenhouse link provided to you via email. Please let us know if you have any questions.

Good luck!
Recruiting @ StockX

Exercise 1: Finding Shoes

At https://jsfiddle.net/xznoa8ue/ there is a series of unit tests that define the behavior of two functions. Write the code that would implement those two functions, making the unit tests pass. Write the code as if you were writing it for production. In addition, write any additional unit tests that you think are needed to specify/validate the function's behavior (use your best judgment to define what the function should do if it's undefined in the existing tests).

`[Insert a link to your saved JSFiddle or a Github Gist]`

Exercise 2: API Design

Joyce Glen is the owner of a world renowned ice cream shop in Beaver Island, Michigan. Joyce is an extremely talented entrepreneur who knows the value of making data-driven decisions. The shop has plans to relocate within the island but they'd like to know what would be a new, optimal location. She asks Stacey, the shop's Chief Data Scientist for her input. After months of research, Stacy concludes there's potential to triple their revenue if they move the shop near the ferry station and build an app which allows arriving visitors to pre-order their ice cream as they wait to deboard the ferry.

Complete the API "spec" by answering the questions below. This API will be used by the shop's app to enable arriving visitors to pre-order their ice cream.

Question #1 - List ice cream flavors

The "spec" below shows the response of an endpoint which lists all available ice cream flavors. The *cost* field in the ice cream payload body has not been defined. Replace `<XYZ>` with some sample data and explain the pros and cons of using the data type you selected.

```
# HTTP Method:
```

```
GET

# Endpoint:
https://api.joyceglenicecream.com/products/

# Request Headers:
Content-Type: 'application/json'

# Request Body:
Not applicable (N/A)

# Response Body:
[
  { "id": 1, "name": "Chocolate", "cost": <XYZ>, "currency": "USD" },
  { "id": 2, "name": "Strawberry", "cost": <XYZ>, "currency": "USD" },
  { "id": 3, "name": "Vanilla", "cost": <XYZ>, "currency": "USD" }
]
```

Question #2 - Place an order

Complete the “spec” below with the elements required to perform a request which will create an order for one of the ice cream flavors available in the previous question

```
# HTTP Method:
<please update as part of answer>

# Endpoint: https://api.joyceglenicecream.com/<please update as part of
answer>

# Request Headers:
Content-Type: <please update as part of answer>

# Request Body:
<please update as part of answer>

# Response Body:
```

Question #3 - Update The Ice Cream Data Entity

The ice cream data entity needs to be updated to support a new backend vendor. Previously the structure of the entity was as follows:

```
{ "id": 1, "name": "Chocolate", "cost": ..., "currency": "USD" }
```

Due to a change in a backend vendor, we can no longer use integers to represent the ice cream ID. The new service requires the ID field to be represented as an UUID.

```
{ "id": "5974...e58a", "name": "Chocolate", "cost": ..., "currency": "USD"
}
```

Please update the existing endpoint (https://api.joyceglenicecream.com/products/) such that that new applications use the UUID datatype of the ID field (as opposed to the original integer datatype). Your change must be backwards compatible given there are older versions of the app being used.

```
# HTTP Method:
<please update as part of answer>

# Endpoint:
<please update as part of answer>

# Request Headers:
<please update as part of answer>

# Request Body:
Not applicable (N/A)

# Response Body:
<please update as part of answer>
```

Question #4 - Performance

After 5 years, Joyce has expanded her business to over 840 locations across the globe and the original API implementation (1 machine running a vanilla Node.js express server) is struggling to scale given the number of requests per minute (RPM).

Write a few paragraphs explaining what changes must be made in order to scale the original API implementation (1 machine running a vanilla Node.js express server) such that the API is more scalable (i.e. able to handle 10MM - 40MM RPM).