the challenge

Our statisticians have developed a fancy new model which assesses how risky a commute is in micromorts, and we'd like to deploy it for validation trials. We need an API that can take in the raw data, feed it to the model, and return the result to the front end. Since users will be uploading their own data in spreadsheets via the front end, we should expect that it might be messy and do some data validation.

model input

The model will all be abstracted into a single function which takes in a dictionary. All we need to do is pass it the correct data, and can assume the model will return a correct result. The model takes in data that's the same structure that's being passed into the API, with a few rules (see below).

model output

The model outputs a single positive integer: that commute's risk in micromorts. We can assume that the number will never be over 1 million and always greater than 0.

API output

We have some discretion in how to structure the result to the front end, as we haven't built that yet. But it should at least contain the ID and the result.

API input

Here are a few examples:

POST body:

```
{
        "commuterID": "COM-1",
        "actions": [{
                "ts": "2022-02-01 8:40:00",
                "action": "drove a car",
                "unit": "mile",
                "quantity": 67
        }]
}
```

The model responds with `10`.

POST body:

```
{
        "commuterID": "COM-42",
        "actions": [{
                        "ts": "2022-01-01 10:05:11",
                        "action": "walked on sidewalk",
                        "unit": "mile",
                        "quantity": 0.4
                },
                {
                        "ts": "2022-01-01 10:16:52",
                        "action": "took a bus",
                        "unit": "mile",
                        "quantity": 12
                },
                {
                        "ts": "2022-01-01 10:30:09",
                        "action": "rode a shark",
                        "unit": "minute",
                        "quantity": 3
                },
                {
                        "ts": "2022-01-01 10:33:45",
                        "action": "took elevator",
                        "unit": "floor",
                        "quantity": 20
                }
        ]
}
```

The model responds with `105124`.

POST body:

```
{
        "commuterID": "COM-64",
        "actions": [{
                        "ts": "2022-01-01 10:05:11",
                        "action": "jogged",
                        "unit": "mile",
                        "quantity": 67
                },
                {
                        "ts": "2022-01-01 10:16:52",
                        "action": "scuba dove through a cave",
                        "unit": "quantity",
                        "quantity": 20
                },
                {
                        "ts": "2022-01-01 10:45:45",
                        "action": "walked up stairs",
                        "unit": "floor",
                        "quantity": 120
                }
        ]
}
```

The model responds with `90`.

data rules

- each request should have a commuterID which is prefixed with `COM-` and followed by a positive integer
- each action should have a timestamp, action, unit, and quantity
- the timestamps should all be on the same day
- actions might be any string
- units should be one of the following: `mile`, `floor`, `minute`, and `quantity`. The statisticians say they'll probably write a v2 model next year with more.

deliverable

- a runnable prototype that accepts POST requests
- tests which demonstrate returning the expected result
- handles input validation
- language agnostic, as long as there are directions for installation
- send it in a zip or put it in a Github gist or code repository