Citrine Informatics Technical Challenge

Backend Engineer

Version: 906bfcb3f72d00bf25f922eed60bcf411049f878

This challenge should take about five hours, both parts included. If you find yourself spending substantially more time than that, consider reducing the complexity of your implementation.

writing

Communication to both technical and non-technical coworkers is critical. For this part of the challenge, pretend that you are living in the hypothetical world described below:

You are an engineer working on the Data Ingestion pipeline. You see developers struggle to release their code, as they sometimes produce bugs or mess up the deployment scripts. You believe that a continuous integration and deployment system would dramatically improve the quality and cadence of engineering work, but the CTO is skeptical when you bring it up:

- We have too many customers to configure CI/CD for - at Citrine every customer gets their own AWS account
- We should release our code at the end of every sprint
- Each team uses different technologies
- We can’t afford to take engineers off of product work to work on it

Write an e-mail to our CTO to convince him that implementing a CI/CD workflow would be a worthy investment.

software

Write a web service in one of the following languages: Java, Python, Ruby, JavaScript. It will perform unit conversion to SI from their “widely used” counterparts. Our tests are automated, so be sure to design your API path exactly as described below. **Please submit code you would be comfortable contributing to a shared codebase. In addition to running automated tests, we will also read your code.**

The web service you will write has a single endpoint with a single method, **Convert Units**, which will convert any properly formatted unit string to their SI counterparts.

Please deliver your code in a way that is easy for us to run. Acceptable choices include a dockerfile, a running service on Heroku, a jar file, etc.

Convert Units

```
METHOD: GET
PATH: /units/si
PARAMS: units - A unit string*
RETURNS: conversion - A conversion object**
```

*A valid unit string is a string containing any number of units multiplied or divided, potentially containing parentheses. Valid examples include `degree` , `degree/minute` , `(degree/(minute*hectare))` , `ha*°` .

**A valid conversion object is a JSON object containing two fields: `unit_name` and `multiplication_factor` . `unit_name` is the unit string from the request, with all units converted to to their SI counterpart. Reducing units is not necessary, so `s/s` is perfectly valid output. The parentheses should match the request unit string. `multiplication_factor` is a floating point number (with 14 significant digits) that you can use to convert any input in the original units to the new SI units.

Example Request:

```
GET /units/si?units=(degree/minute)
```

```
{
    "unit_name": "(rad/s)",
    "multiplication_factor": .00029088820867
}
```

Unit Conversion Factors

This table denotes valid input and conversion factors for you to implement. Either values from the left two columns are valid input, as are SI units themselves.

Name	**Symbol**	**Quantity**	**SI Conversion**
minute	min	time	60s
hour	h	time	3600s
day	d	time	86400s
degree	°	unitless/plane angle	(π/180) rad
arcminute	'	unitless/plane angle	(π/10800) rad
arcsecond	"	unitless/plane angle	(π/648000) rad
hectare	ha	area	10000 m^2
litre	L	volume	0.001 m^3
tonne	t	mass	1000 kg