Engineering Challenge On-Site

Challenge Prompt

Create a system that extracts menu items from a few arbitrary restaurants on a food delivery platform, and posts them into an online database via a RESTFul API. The system should get the list of menu items (dishes), and then hit the POST items API to save them.

The process of listing menus and posting data into the storage **must be** asynchronous. It means you should make requests to obtain menu data for all the restaurants in parallel. Once results for all restaurants returned, start adding those concurrently into the storage.

You can use any programming language, framework, and IDE to demonstrate your best work; *however*, we ***strongly discourage*** the use of microservices, kafka, DBs, etc.

Getting started

Your program should accept a list of arbitrary restaurant IDs.
You are provided with a personal API endpoint for storing menu items which can be obtained using the following link: https://menu-processor-api.fly.dev/docs
You can use it to list or delete the data you've created. We will also use it to verify the results of the challenge.

Get menu items from restaurants

You should extract the menu items (dishes) from the provided API for a few different restaurants.
You can get a list of restaurant IDs using the same API collection, for example:
b2bi, v6zw, w7cg.

Saving menu items to the storage

You should use the provided API endpoint to post the fields below for each item you create:

- restaurant
- rating
- category
- item_id
- item_name
- item_price

Make sure to use the same `user_id` for all API calls (any string of your choice).

Grading Rubric

- ❏ Meets all requirements from the *Challenge Prompt*
- ❏ Is valid, runnable code *(via CLI or IDE)*
- ❏ Has appropriate usage of design patterns, concurrency, and data structures
- ❏ Has comprehensive unit testing
- ❏ Has console output that allows **interviewers to *understand your system's operation* as it runs in real-time**. Output events (list menus, post each item into the storage) as they occur so the operation of your system can be understood as it runs
- ❏ Has production-quality code cleanliness
- ❏ Has production-quality docs on all public functions *(as if someone had to work with your code)*
- ❏ Has a README file that contains
    - ❏ Instructions on how to run and test your code in a local environment
    - ❏ Any other design choices you would like the interviewers to know

Submission

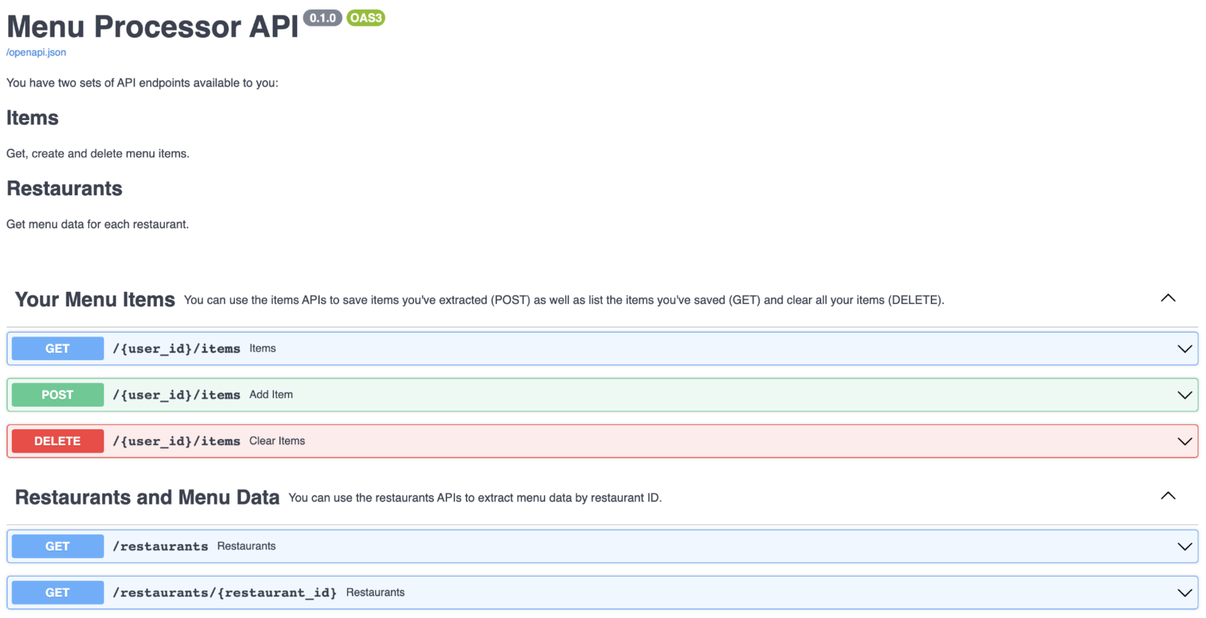
Please compress/zip your code and submit it to us via a GreenHouse link. You can find the link in your email with the subject "Coding Challenge from CSS".

Please do not place your submission on a public facing repository, such as GitHub/Bitbucket/etc.

Appendix

Testing the API Endpoints

1. Visit https://menu-processor-api.fly.dev/docs
2. Choose an endpoint
3. Click "Try it out" and enter your `user_id`
4. Click "Execute"


Menu Processor API 0.1.0 OAS3

/openapi.json

You have two sets of API endpoints available to you:

Items

Get, create and delete menu items.

Restaurants

Get menu data for each restaurant.

Your Menu Items You can use the items APIs to save items you've extracted (POST) as well as list the items you've saved (GET) and clear all your items (DELETE).

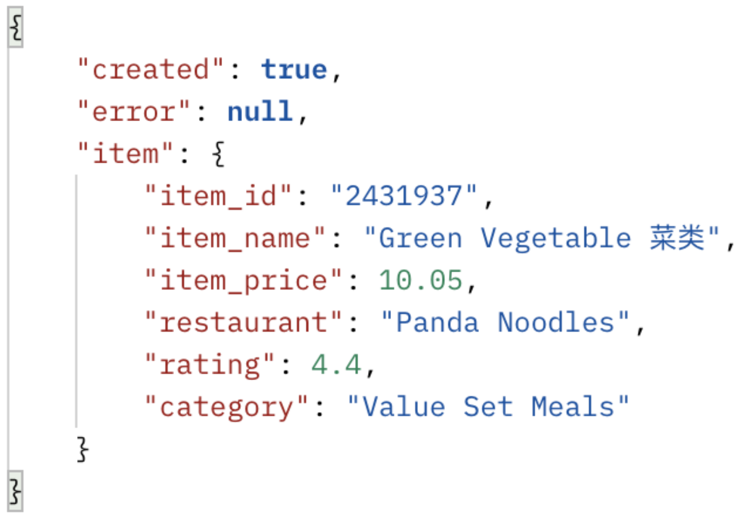
```
GET     /{user_id}/items  Items
POST    /{user_id}/items  Add Item
DELETE  /{user_id}/items  Clear Items
```

Restaurants and Menu Data You can use the restaurants APIs to extract menu data by restaurant ID.

```
GET     /restaurants  Restaurants
GET     /restaurants/{restaurant_id}  Restaurants
```

You are expected to make the endpoint URL (or user ID) configurable in the code so we can test your system with different endpoints.

You will be using **POST** to create items.
When you successfully created an item you should receive a **200** response with `{...“created”: true}`:



```
{
    "created": true,
    "error": null,
    "item": {
        "item_id": "2431937",
        "item_name": "Green Vegetable 菜类",
        "item_price": 10.05,
        "restaurant": "Panda Noodles",
        "rating": 4.4,
        "category": "Value Set Meals"
    }
}
```

Log Format

Output Events

```
Getting Menu: v6zw
Getting Menu: w7cg
Loaded menu: v6zw - Viet Thai Cuisine (Toa Payoh)
Adding item 1995889 (A1. Pad Thai Value Set) from v6zw to the storage
Adding item 1995890 (A2. Thai Spicy Fried Kway Teow Value Set) from
v6zw to the storage
Loaded menu: w7cg - Viet Thai Cuisine (Toa Payoh)
Added item 1995890 (A2. Thai Spicy Fried Kway Teow Value Set) from
Adding item 2431935 (Green Vegetable 菜类) from w7cg to the storage
Added item 1995889 (A1. Pad Thai Value Set) from v6zw to the storage
Adding item 1995891 (A3. Thai Style Fried Vermicelli Value Set) from
v6zw to the storage
Added item 1995891 (A3. Thai Style Fried Vermicelli Value Set) from
v6zw to the storage
Added item 2431935 (Green Vegetable 菜类) from w7cg to the storage
```

Output Format

Once your application finishes running, you can print out the table of all the items you've saved, including whether it was successful or not.

Output Data

```
Saved 2 items, 1 failure:
---
Restaurant, Category, Item, Success
---
Viet Thai Cuisine, Value Set Meals, A1. Pad Thai Value Set, true
Viet Thai Cuisine, Value Set Meals, A2. Thai Spicy Fried Kway Teow
Value Set, false
Panda Noodles, Vegetarian Meals, Green Vegetable, true
```

Extra Credit

Do these if only you've completed all the aspects of the challenge under the Grading Rubric (including comprehensive test coverage).

1. Try to make all aspects of the application truly asynchronous, for example, start saving items as soon as one restaurant response is returned rather than collecting responses from all restaurants first.
2. Try to extract menu items from the actual FoodPanda website. We welcome creative ways to extract the menu information from the site.