Paribus

Paribus Python Challenge

Hi! We're very excited that you're interviewing at Paribus. We created this task so that we can evaluate your ability to learn and code. We hope that you learn something new working on this. Good luck! ☺️

Task: Hospital Directory API

You are tasked with building a RESTful API for a basic Hospital Directory application. This API will allow users to manage hospital information, including adding, viewing, updating, and deleting hospital records.

Estimated Completion Time: 2-4 hours

Functional Requirements

Your API should support the following operations for hospitals:

- **Add Hospital**
  - Include `creation_batch_id` to create hospital as part of a batch and initialize it as inactive
  - Without batch ID: Hospital is created as active
- **Get all Hospitals**
- **Get Hospital by ID**
- **Update Hospital**
- **Delete Hospital**
- **Import from CSV**
  - Accept a CSV file containing hospital records
  - Generate a unique UUID to use as the batch ID for all hospitals in the CSV
  - For each row: validate required fields and add the hospital to the system with the batch ID if valid
  - All hospitals from CSV import should start as inactive (due to batch ID)
  - Return a summary of successes and failures for each row, including the batch ID

    Valid CSV format example:

```
name,address,phone
Sunrise Hospital,123 Main St,+1-555-1111
Green Valley Clinic,456 Elm St,+1-555-2222
```

Batch Operations:

- **Get Hospitals by Batch ID**
- **Delete all Hospitals in a Batch**
- **Activate all Hospitals in a Batch**
  - Only succeeds if ALL hospitals in the batch are currently inactive
  - Throws error if any hospital in the batch is already active

Hospital Model: Each hospital should have at least the following attributes:

- `id` (unique identifier, e.g., integer)
- `name` (string)
- `address` (string)
- `phone` (string, optional)
- `creation_batch_id` (UUID, optional - for batch processing)
- `active` (boolean - false when batch_id provided, true otherwise)
- `created_at` (timestamp, automatically set on creation)

Optional Tasks (Bonus Points)

If you have extra time or want to showcase additional skills, consider implementing one or more of the following:

- **Pagination:** Implement basic pagination for the Get all Hospitals endpoint
- **Database Persistence:** Persist the data using SQLite or similar lightweight database
- **Basic Error Handling:** Implement more robust error handling for invalid input or server errors
- **Unit Tests:** Write unit tests for your API endpoints or core logic using `unittest` or `pytest`
- **Dockerization:** Provide a `Dockerfile` and `docker-compose.yml` to easily run your application in a Docker container

Expected Tech Stack

- **Language:** Python
- **Web Framework:** FastAPI or Flask (preferred for its minimalism)
- **Data Persistence:** In-memory storage is acceptable

- **Deployment:** Deploy the app online. Suggested platforms: Render

Evaluation Rubric / Checklist

Your submission will be evaluated based on the following criteria:

- **Functional Correctness (40%):** All specified API endpoints work as described
- **Deployment (20%):** App is running online and link is shared
- **Code Quality & Readability (10%):** Clean, readable, well-organized code
- **Data modeling and API Design (10%):** Proper data structure, minimal validation where needed
- **Documentation (10%):** Clear README with instructions and example usage
- **Optional tasks (10%):** Any optional tasks completed and working correctly

Submission Instructions

Please include the below in your assignment submission:

- A Git repository (e.g., GitHub, GitLab, Bitbucket) including all source code
- A public URL for the hosted application

Good luck, and we look forward to reviewing your work!