PROJECT: FoodieConnect — Your Data Analyst Internship Begins

Story

You've just joined **FoodieConnect**, a startup that helps users find the best food places in their city.

Your manager assigns you tasks to analyze **users, restaurants, and their ratings** — using **pure Python (no pandas, no numpy)**.

This project mirrors the structure of *Coders of Delhi* but is **simpler**, with fewer data relationships and easier logic.

🍕 TASK 1 — Load and Explore Data

Dataset:
Users can give ratings to restaurants. Restaurants have categories (pizza, cafe, biryani, etc.)

Example JSON:

```
{
 "users": [
  {"id": 1, "name": "Aman", "rated": {"201": 5, "202": 3}},
  {"id": 2, "name": "Riya", "rated": {"202": 4}},
  {"id": 3, "name": "Kabir", "rated": {"203": 2}}
 ],
 "restaurants": [
  {"id": 201, "name": "Pizza Town", "category": "Pizza"},
  {"id": 202, "name": "Cafe Brew", "category": "Cafe"},
  {"id": 203, "name": "Biryani Hub", "category": "Biryani"}
 ]
}
```

Your Task

Load this JSON file

Print:

All users with ratings

All restaurants

Super easy compared to Coders of Delhi.

🧹 TASK 2 — Clean the Data

Problems in the messy dataset:

Missing user names

Ratings with values outside 1–5

Restaurants with duplicate IDs

Example messy issues:

User with empty name

Duplicate restaurant

Rating = 7 (invalid)

Your Task:

Write cleanup steps:

Remove users with missing names

Fix out-of-range ratings (keep only 1 to 5)

Deduplicate restaurants based on ID

This is **much easier** than cleaning friend lists + duplicate pages.

🤝 TASK 3 — Find "Restaurants You May Like" (Easy Recommender)

Instead of mutual friends → **shared taste in food**.

Logic:

If two users rated the same restaurant

Recommend restaurants the other user liked.

Example:

Aman rated Pizza Town (5)
Riya rated Cafe Brew (4) AND Pizza Town (3)

→ Suggest **Cafe Brew** to Aman.

Your Task:

Implement simple collaborative filtering:

If two users rated the same restaurant,

recommend the restaurants the other user rated 4 or 5 stars.

No mutual friend logic → simpler.

⭐ TASK 4 — Find "Top Restaurants" (Easy Aggregation Task)

Compute:

Average rating of each restaurant

Sort restaurants by rating

Beginner-friendly: just loops + dict.

🥇 Bonus (Optional + Very Easy)

Find:

- Most active user (maximum ratings)

  Most popular category (Pizza, Cafe, etc.)