Aktos Engineering Take Home

Background

Write a Django application that allows a collection agency to ingest data files provided by their clients.The data file is a single CSV file [CSV](https://drive.google.com/file/d/1W_kilVANsPW3tUUFLPOB7WEibvNItVJh/view?usp=sharing) containing account information about consumers and what they owe to a single client. The accounts here can have many consumers that owe the debt, and the consumers can have many accounts.

Client: Organization that hires a collection agency to collect debt on their behalf. A collection agency can work with many clients to collect debt.
Consumers: The person(s)/entities that owe the debt.
Debt: The amount owed by one or more consumers.

Assuming all the consumers belong to the same collection agency and same client, create a system to handle the ingestion of the data, and retrieval of the data.

Requirements

Create an endpoint '/accounts' that finds all the accounts for a collection agency and their clients.

Sample API requests:
1. `GET /accounts?min_balance=100&max_balance=1000&status=in_collection`
2. `GET /accounts?min_balance=100.23&status=collected&consumer_name=john`

Allowed parameters to the endpoint:
1. **min_balance**: The min amount to filter the data by.
2. **max_balance**: The max amount to filter the data by.
3. **consumer_name**: A query for the consumer name to filter the data by.
4. **status**: The status of the debt to filter the data by.

All query parameters are optional, and the minimum and maximum fields should be inclusive (e.g. min_balance=2&max_balance=3 should return consumers that have balances between 2 and 3 inclusive).

Minimum Requirements:

- Must be a Django app
- Appropriate modeling of the data
- Ingestion of the data
- API endpoint(s) to return a list of accounts (/accounts)
- API endpoint(s) to filter results correctly by any combination of API parameters
- Usage of a datastore
- Appropriate testing (as you see fit).

Bonus:

- Read the CSV through an endpoint
- Pagination of the APIs (describe the pagination you used and what are the pros/cons for going with the type of pagination you picked)
- Allow modeling for many agency and many clients

Submission Checklist

- Deploy this somewhere (e.g Heroku, AWS)
- A video recording of you walking through the code along with what you would do if you were given more time and/or any design decision you had to make.
- When you're done, please share the project with 'itsdpao' and 'akif-hossain' on Github and reply the email cc'ing daniel@aktos.ai with links to the deliverables.

Evaluation Criteria

- Appropriate modeling of the data
- Correctness in filtering logic for the API
- Future maintainability of the code while keeping time constraints in mind
- Consistency in code styling/guidelines (we use black but feel free to use any other as you see fit)
- Appropriate test coverage and documentation in code (usage of TODO/NOTE)

For any questions, you can reach out via email.