Backend Engineer / Technical Assignment

Submission:

Your code should be submitted by providing us a link to a **private** repository (e.g. Github), with clear instructions on how to run it.

Please make sure to add the following users as collaborators, so we can view your submission:

- asamokhina
- flynndcs
- senning

Task Description:

In this task, you are requested to build a Backend/Data application.

You are provided with some raw input data (see included file). The file contains rows of usage data, each with some metadata columns plus a geometry column.

The application will offer 2 main functionalities:

1. Trigger a data processing operation to transform the raw data into usable data saved on a DB.
2. API server.

Usability Requirements:

- The user should be able to connect to the API endpoints and perform operations.
- The result of the ETL operation should be saved in a db.

Technical Requirements:

- In addition to the code, we would like you to submit this application in a docker compose file, containing all the necessary components to run both the application and the test suites. Provide clear explanations on how to run the code you submit.
- We recommend that the data processing operation will be done during the startup of the docker compose environment. However, if you think of a better approach that you want to implement and you think is more reasonable, please feel free to do it. We will expect you to explain the rationale behind your choices.
- The backend should be created with a Python framework (such as FastAPI), or with a Go framework (Such as GIN). A solution in either of the 2 languages is acceptable.
- Please include appropriate test coverage for your code.

- Finally, documentation will play a big part in how we assess your solution, so please make sure that you document each of the components, and the system as a whole.
- Please provide a mini-design (a simple diagram should work) of how the CI/CD pipeline for such a project will look like. This pipeline should be triggered whenever a new commit is pushed to a branch.
- **BONUS:** Provide a brief outline for how you would add authentication for your API. Assume that users will have accounts, and some of the routes in this project will only be available to authenticated users.
- For specific requirements, please refer to the descriptions below.

Data Processing Requirements:

We have a file that contains data in a table with three columns: org_id, footprints_used, and source_event_timestamp.

- org_id is a unique identifier for an organization that generates usage data at a particular time.
- footprints_used is a type of data that describes a geographic area with a specific shape, size, and location.

For the purpose of generating a heatmap in a frontend application (out of scope for this task), we need to sanitize the data. Our task is to prepare this data for use in a user interface (UI) by following these steps:

- Extract the necessary information from the CSV.
- Transform the data
- Load the transformed data into a database that can be used to support the UI.

Use any database that you feel comfortable with (but if you want a smooth ride, give PostgreSQL/PostGIS a try)

You can use any Python or Golang library that you like, so feel free to pick the one that works best for you! Some sweet examples of popular libraries include Shapely, GeoPandas, GeoGO, and go-geom.

Bonus :
The footprints/geometries used in Planet can be trickier due to their size and number of vertices. Suggest or implement how to handle these complex geometries in the current assignment.
You might find some examples in the attached CSV.

Backend Requirements:

On the API, endpoints should be created to support the following actions:

- An endpoint which returns the data in a form that the frontend can render into a heatmap. Assume that the frontend uses Mapbox to create the map - your endpoint should return a geojson FeatureCollection with a list of Points representing the features in your data.
- An endpoint to fetch org_ids of all the org ids that have usage data.

We highly appreciate your time and effort.

Thank you!