👉 Our Expectations

We greatly appreciate your commitment to this process.

The main goal of this task is to:

- Get to know how you work
- Assess your general understanding and expertise in relevant aspects of backend development
- Ultimately, have a conversation with you to discuss the actions you performed, results, and any issues encountered.

We do not want to create a stressful situation for you or interfere with your daily activities. Therefore, you should not invest more than a few hours in executing this task.

Please feel free to contact us at any time if you have any questions.

We are confident that you will do your best, and we are looking forward to reviewing the output with you.

💡 Track and Trace API

The Track and Trace page enables end users (receiver of a shipment) to monitor the status of their shipment.

Your task is to create a backend API that serves shipment data and provides users with current weather conditions at their location.

Guideline

You have been provided with a CSV file (below) containing sample shipment and article data to seed your data structures. Your task is to create an API application that performs the following:

- Provides a RESTful or GraphQL Endpoint that exposes shipment and article information along with corresponding weather information.
- Allows consumers to lookup shipments via tracking number and carrier.

Hints:

- Integrate a suitable weather API: Choose a weather API (e.g., OpenWeatherMap, Weatherbit, or any other free API) and fetch the weather information for the location of each shipment.
- Limit weather data retrieval: Ensure that weather information for the same location (zip code) is fetched at most every 2 hours to minimize API calls.
- You must use Django, and you can use any library that you feel comfortable with.

Nice to have:

- Provide unit tests and/or integration tests for the application.
- OpenAPI docs

Solution

Deliverables:

- Application code, tests, and documentation(could be Persian) needed to run the code.

Evaluation Criteria:

- Code quality and organization
- Functional correctness
- Efficiency, robustness, and adequate design choices

Discussion Points:

- What were the important design choices and trade-offs you made?
- What would be required to deploy this application to production?
- What would be required to scale this application to handle 1000 requests per second?

Seed Data

NB: you don't need to write an import, if that is too time consuming, you can just setup one or two sample items in the DB via any mechanism you like.

```
tracking_number,carrier,sender_address,receiver_address,article_name,articl
e_quantity,article_price,SKU,status
TN12345678,DHL,"Street 1, 10115 Berlin, Germany","Street 10, 75001 Paris,
France",Laptop,1,800,LP123,in-transit
TN12345678,DHL,"Street 1, 10115 Berlin, Germany","Street 10, 75001 Paris,
France",Mouse,1,25,MO456,in-transit
TN12345679,UPS,"Street 2, 20144 Hamburg, Germany","Street 20, 1000
Brussels, Belgium",Monitor,2,200,MT789,inbound-scan
TN12345680,DPD,"Street 3, 80331 Munich, Germany","Street 5, 28013 Madrid,
Spain",Keyboard,1,50,KB012,delivery
TN12345680,DPD,"Street 3, 80331 Munich, Germany","Street 5, 28013 Madrid,
Spain",Mouse,1,25,MO456,delivery
TN12345681,FedEx,"Street 4, 50667 Cologne, Germany","Street 9, 1016
Amsterdam, Netherlands",Laptop,1,900,LP345,transit
TN12345681,FedEx,"Street 4, 50667 Cologne, Germany","Street 9, 1016
Amsterdam, Netherlands",Headphones,1,100,HP678,transit
TN12345682,GLS,"Street 5, 70173 Stuttgart, Germany","Street 15, 1050
Copenhagen, Denmark",Smartphone,1,500,SP901,scanned
TN12345682,GLS,"Street 5, 70173 Stuttgart, Germany","Street 15, 1050
Copenhagen, Denmark",Charger,1,20,CH234,scanned
```

That is it - everything else is up to you! Happy coding!