Test job

Lead time

- environment preparation - no more than 10 minutes (configuration is performed automatically through the prepared docker-compose file, see instructions below)
- task performance - no more than an hour (performance time depends on the candidate's level);
    - data validation is not required
    - using ORM to execute database queries is not required, you can use `DB::select`

Preparing for the task (setting up the environment)

For your convenience, docker-compose.yml file is prepared for quick deployment of the environment with already prepared PostgreSQL database and an empty Laravel project. To run it, you need to have docker software installed.

In the .env file you can configure the following parameters if necessary (for example, if ports 7777 and/or 5555 are already in use):

- **PORT** - the port number on which the site will be accessed (by default 7777, i.e. after starting the server in a browser, the site can be opened at the address http://localhost:7777/)
- **DB_PORT_EXTERNAL** - the PostgreSQL DBMS port, through which you will be able to connect to if you want to work with the database directly, for example, through DBeaver or other software (by default port 5555)

In the command line in the directory with the project, run

```
docker-compose up
```

Warning: *the first run will download the necessary docker images as well as dependencies via Composer; keep this in mind if you have a limited internet connection with limited traffic.*
internet connection with limited traffic.

Source data

The database contains three tables:

- aircrafts - directory of aircraft numbers
- airports - directory of airports
- flights - the main table in this task
    - aircraft_id - id of the aircraft number

    - airport_id1 - id of the departure airport
    - airport_id2 - id of the boarding airport
    - takeoff - departure time
    - landing - landing time
    - cargo_load - loading volume at the departure airport
    - cargo_offload - the volume of unloading at the airport of landing

The table structure can also be viewed directly in the database, or in the script from which it is automatically initialized when deployed via **docker-compose**:

entrypoint/database:

- 010-db_init.sql - creating tables
- 020-aircrafts.sql - filling in the flight number table with test data
- 030-airports.sql - filling in the airport table with test data
- 040-flights.sql - filling in the flight table with test data

Setting the task

Implement the endpoint API:

- HTTP method: GET
- URL: /api/aircraft_airports
- Parameters
    - `tail` - aircraft tail number
    - `date_from` - beginning of the period (format: `yyyy-mm-dd hh:mm`)
    - `data_to` - end of the period (format: `yyyy-mm-dd hh:mm`)

Endpoint returns in JSON format an array of airports where the selected aircraft was located for a specified period, indicating:

- airport_id - airport ID
- code_iata - airport IATA code
- code_icao - airport ICAO code
- cargo_offload - unloading volume at this airport
- cargo_load - loading volume at this airport
- landing - boarding time at this airport
- takeoff - departure time from this airport

Example

URL: `http://localhost:7777/api/aircraft_airports?tail=TEST-001&date_from=2023-01-01%2022:00&date_to=2023-01-02%2015:00`

The result of the call:

```
[
    {
        "airport_id": 44225,
        "code_iata": "BHA",
        "code_icao": "SESV",
        "cargo_offload": 0,
        "cargo_load": 10,
        "landing": "2023-01-01 20:54:28",
        "takeoff": "2023-01-01 23:36:34"
    },
    {
        "airport_id": 29686,
        "code_iata": "COA",
        "code_icao": "KO22",
        "cargo_offload": 0,
        "cargo_load": 90,
        "landing": "2023-01-02 04:00:37",
        "takeoff": "2023-01-02 09:06:33"
    },
    {
        "airport_id": 23767,
        "code_iata": "DKR",
        "code_icao": "GOOY",
        "cargo_offload": 80,
        "cargo_load": 0,
        "landing": "2023-01-02 14:06:49",
        "takeoff": "2023-01-02 15:43:51"
    }
]
```


For understanding, this answer can be interpreted as follows:

The `TEST-001` board stayed at three airports from `01.01.2023 22:00` to `02.01.2023 15:00`:

1. BHA/SESV - landed here on 01.01.23 at 20:54, not unloaded on arrival, extra loaded with 10 tons of cargo, took off to the next airport on 01.01.23 at 23:36
2. COA/KO22 - landed here on 02.01.23 at 04:00, not unloaded on arrival, extra loaded with 90 tons of cargo, took off on 02.01.23 at 09:06
3. DKR/GOOY - landed here on 02.01.23 at 14:06:49, 80 tons of cargo was unloaded, no extra loading, took off on 02.01.23 at 15:43.