Arcadia coding challenge

Daniel Núñez Álvarez

Agenda

Objectives

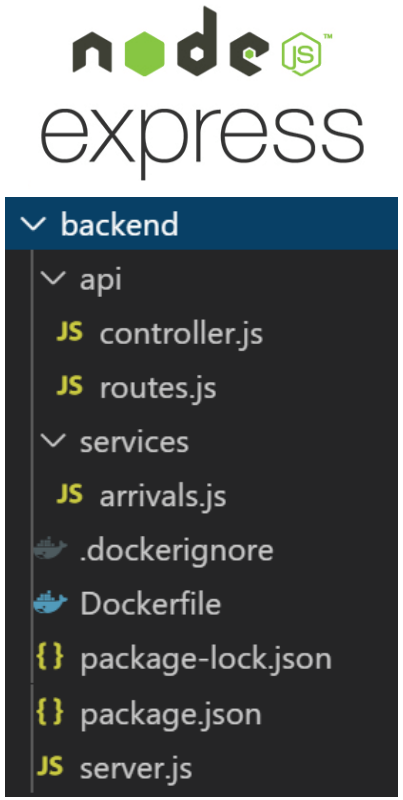
The objective of this challenge is to deploy a full-stack application that reads data from the OpenSky Network API and visualize it in a web user interface.

The challenge is split in three steps:

1. Backend: Create a microservice that provides a connection to an external Restful API.
2. Frontend: Provide a web user interface that visualizes the data and allows the user to interact with it.
3. DevOps: Save the code in a Git repository and provide a ‘docker-compose.yml’ file that automatically build and run the application.

Backend

The backend solution consists in a Restful API Microservice developed using Node JS and Express.



1. The frontend UI calls the API endpoint (/api/v1/opensky/stats/:airport/:begin/:end) specified in ‘route.js’.
2. The API endpoint makes a GET request to the function “getArrivals” defined in ‘controller.js’.
3. The function “getArrivals” calls the function “getArrivalsAPI” defined in the service “arrivals.js”.
4. The function “getArrivalsAPI” makes the request to the external API provided by OpenSky Network.
5. The data is sent back to the frontend UI.

Backend

Frontend UI calls API

route.js

routes.js

backend > api > routes.js > ...

```
1  //Definition of the routes
2
3  'use strict';
4
5  const controller = require("./controller");
6
7  module.exports = function (app) {
8      app.route("/api/v1/opensky/stats/:airport/:begin/:end")
9      .get(controller.getArrivals);
10 };
```

controller.js

controller.js

backend > api > controller.js > ...

```
1  //Definition of the controller
2
3  'user strict';
4
5  const arrivals = require('../services/arrivals');
6
7  var controller = {
8      getArrivals: function(req,res){
9          arrivals.getArrivalsAPI(req,res, function(err, data){
10             if(err)
11                 res.send(err)
12             res.json(data)
13         }
14
15         );
16     },
17 };
18
19 module.exports = controller;
```

arrivals.js

arrivals.js

backend > services > arrivals.js > ...

```
1  //Definition of the services
2
3  'user strict';
4
5  const { json } = require('body-parser');
6  const request = require('request')
7
8  var arrivals = {
9
10     getArrivalsAPI: async function (req, res, next) {
11
12         request('https://opensky-network.org/api/flights/arrival?airport='+req.params.airport+'&begin='+req.params.begin+'&end='+req.params.end,
13
14         function (error,response) {
15             if (!error && response.statusCode == 200){
16                 var data={};
17                 var body = response.body
18                 body = JSON.parse(body)
19                 data['arrivals']=body.map(item=> {return {departureAirport:item.estDepartureAirport, callsign:item.callsign, departureAirportHorDistance:item.estDepartureAirportHorizDistance}})
20                 res.send(data);
21                 console.log(data)
22             }
23             else {
24                 console.log(response.statusCode + response.body)
25                 res.send("Error");
26             }
27         })
28     }
29 };
30
31 module.exports = arrivals
```

Return data to frontend UI

Frontend

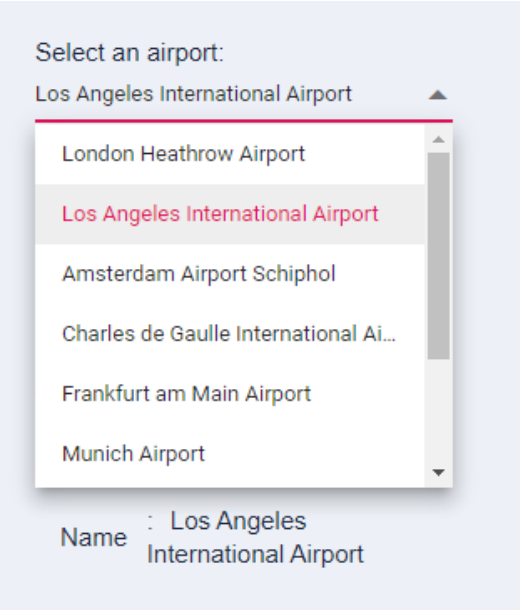
The frontend solution consists in a web application developed using Angular framework together with HTML, CSS and JavaScript.

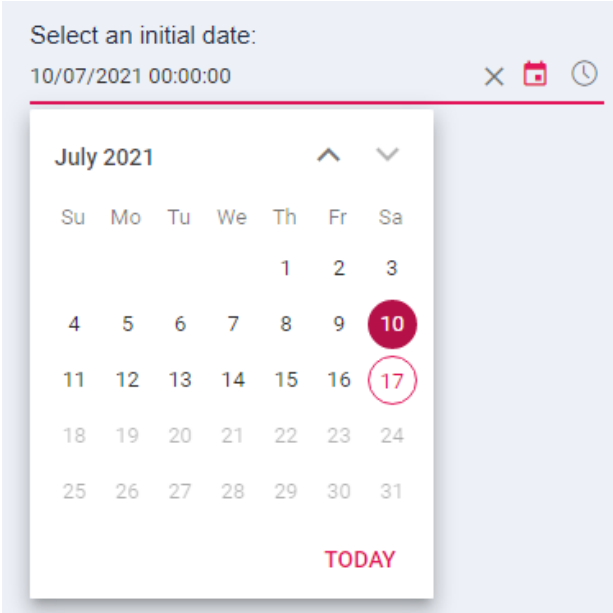
The following features are implemented:

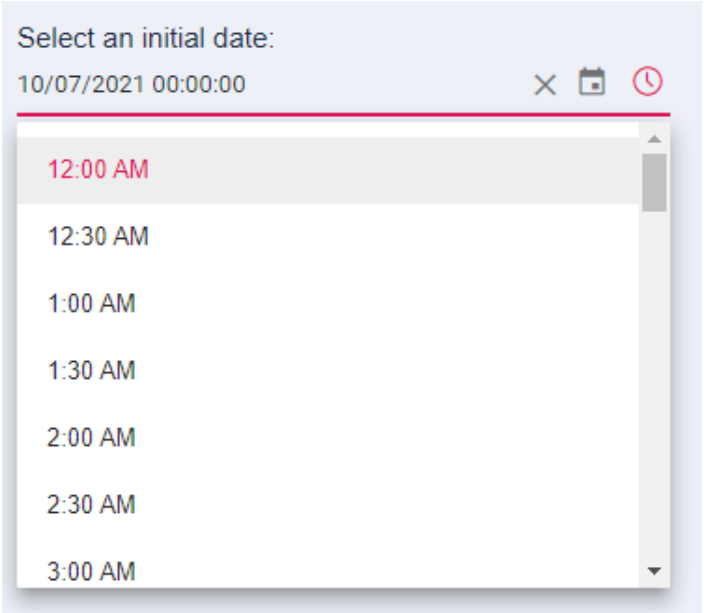
1. A dropdown list with available airports and a date picker to select begin date and end date.
2. A table that shows the results for the given inputs. The table allows pagination, sorting and hide/show columns.
3. A map image showing the location of the arrival airport and the departure airports.

Frontend

User inputs



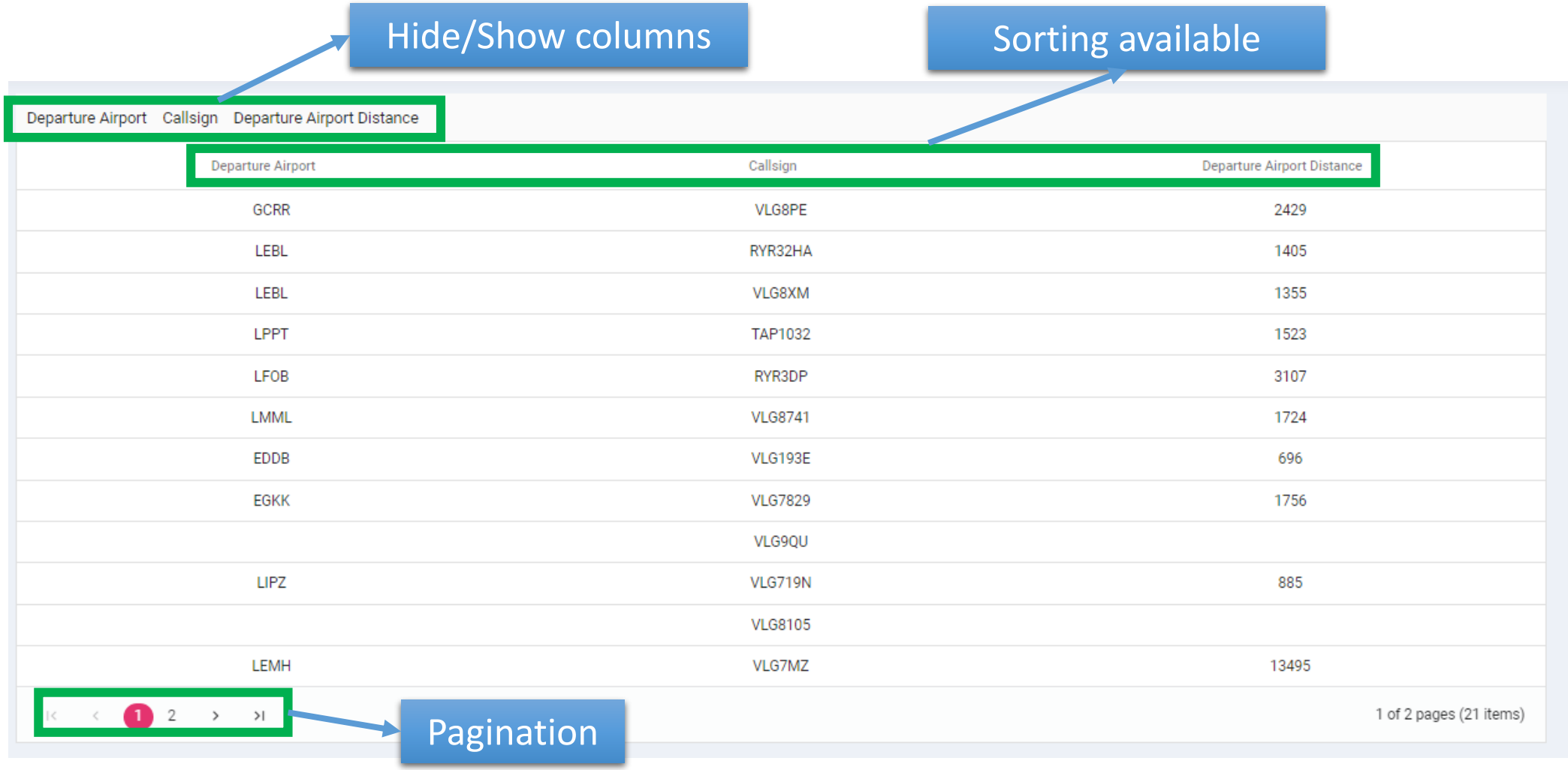




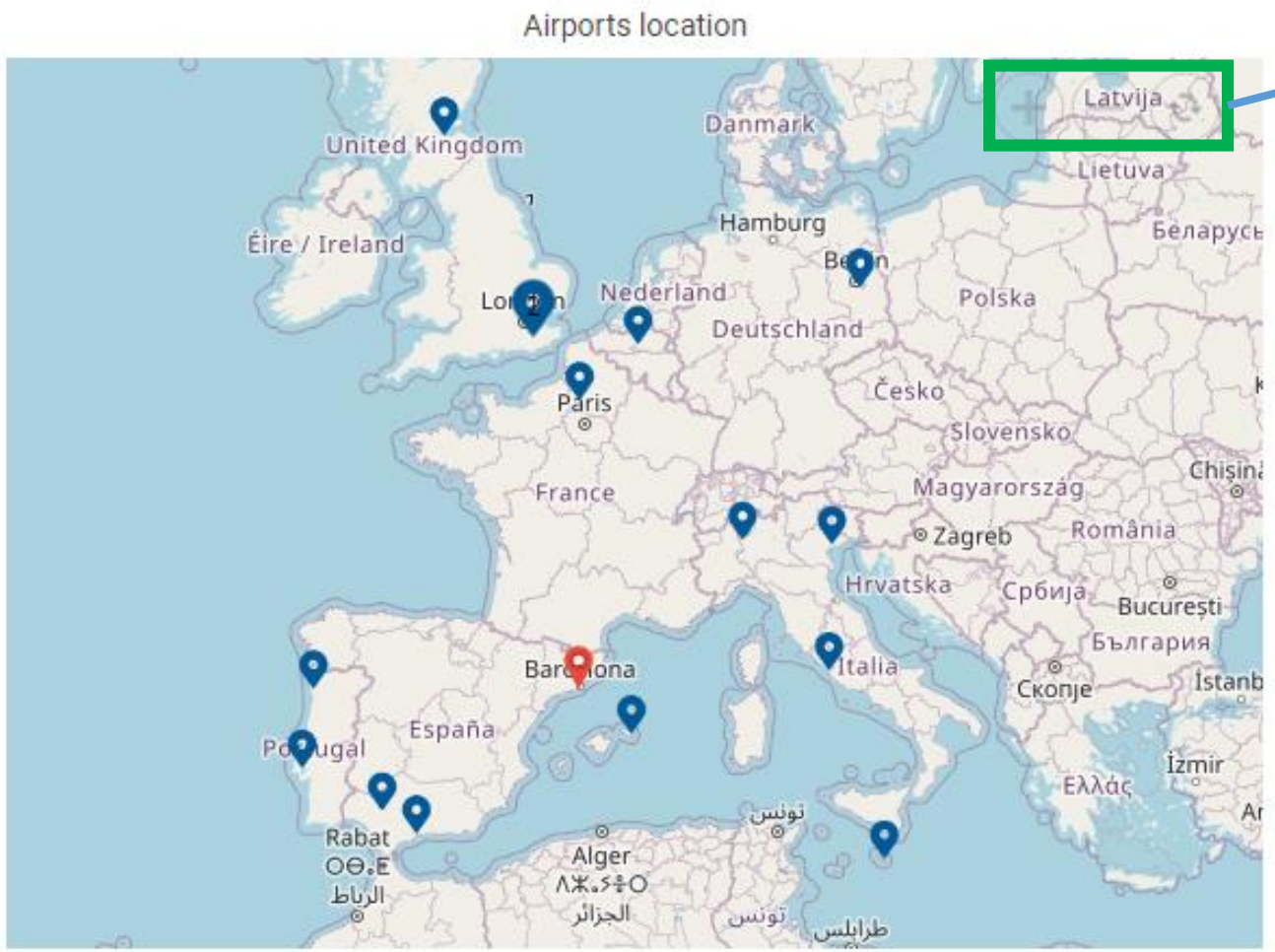
Airport

Initial and end date and time

Frontend


Hide/Show columns
Sorting available
Departure Airport Callsign Departure Airport Distance
Departure Airport
Callsign
Departure Airport Distance
GCRR VLG8PE 2429
LEBL RYR32HA 1405
LEBL VLG8XM 1355
LPPT TAP1032 1523
LFOB RYR3DP 3107
LMML VLG8741 1724
EDDB VLG193E 696
EGKK VLG7829 1756
VLG9QU
LIPZ VLG719N 885
VLG8105
LEMH VLG7MZ 13495
1
2
1 of 2 pages (21 items)
Pagination

Frontend



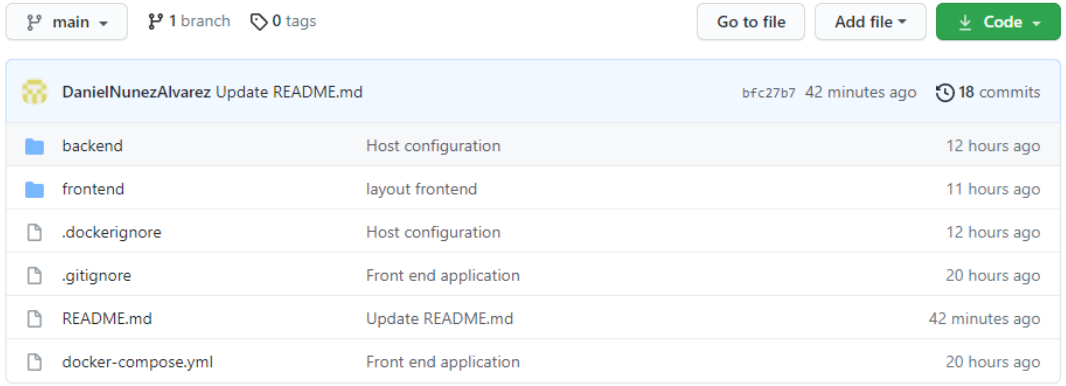
Zoom In/Out

Arrival airport

Departure airports

DevOps

GitHub repository



https://github.com/DanielNunezAlvarez/opensky.git

docker-compose.yml

```yaml
version: '3'
services:
    nodejs-server:
        build:
            context: ./backend
            dockerfile: Dockerfile
        ports:
            - "8080:8080"
        container_name: node-api
        volumes:
            - ./backend:/usr/src/app/backend
            - /usr/src/app/backend/node_modules
    angular-ui:
        build:
            context: ./frontend
            dockerfile: Dockerfile
        ports:
            - "4200:4200"
        container_name: angular-ui
        volumes:
            - ./frontend:/usr/src/app/frontend
            - /usr/src/app/frontend/node_modules
```

Next steps

- Implement authentication method in the backend microservice and the frontend user interface.
- Improve interactive map (Performance with long array results, add aircraft position).
- Improve test implementation.
- Adapt the application to a Progressive Web Application (PWA).
- Additional features and API calls (Flights by Aircraft, Departures for Airport, Track by Aircraft, ...).