influxdata
Act in Time

GEO Visualisation

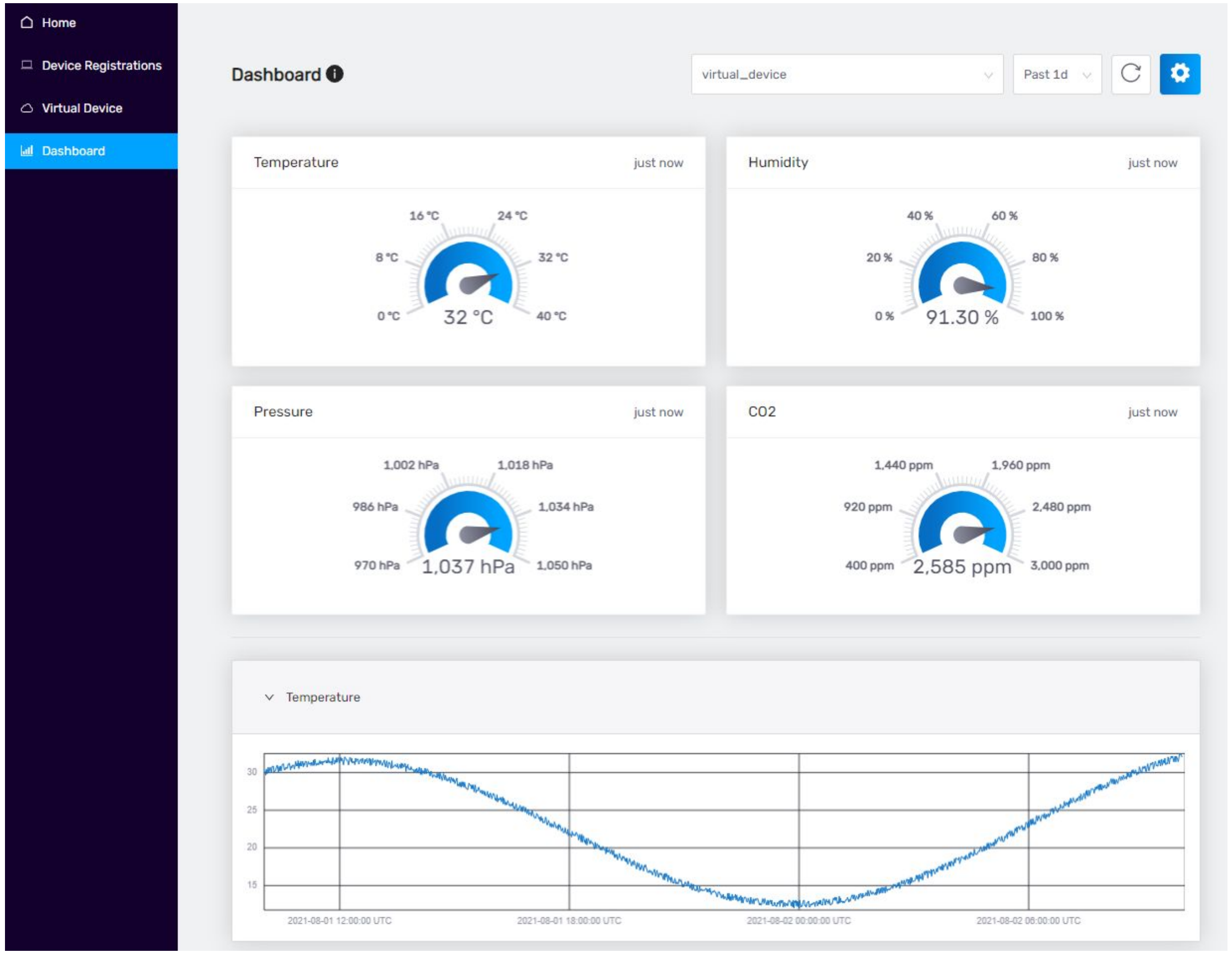
Workshop

Extend Dashboard

Latest Value
- Gauge

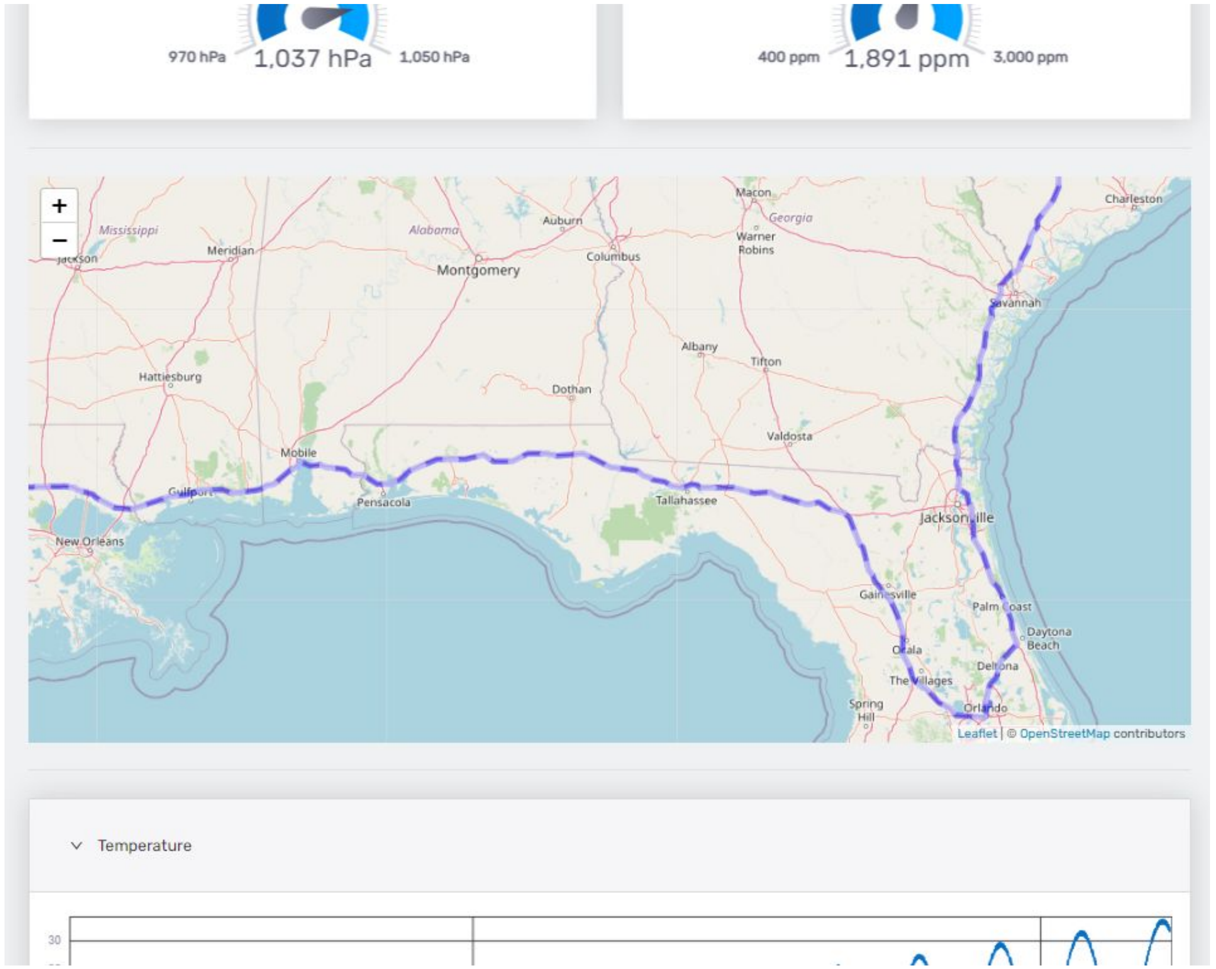
History
- Line chart

Location history
- This exercise



Goal

Insert location history of the IoT device into the dashboard



Steps

All the changes in the dashboard file:

*iot-center-v2\app\ui\src\pages\**DashboardPage.tsx***

1. Update **query** (from explorer exercise)
2. Add GEO **visualisation**

   2.1. Add imports

   2.2. Create visualisation component

      2.2.1. Convert input data

   2.3. Add GEO visualisation into dashboard

1 Update Query

In function **fetchDeviceMeasurements** update flux query:

- fetch GPS fields as well (add highlighted code)

```
const fetchDeviceMeasurements = async (
  config: DeviceConfig,
  timeStart = '-30d'
): Promise<GiraffeTable> => {
  const {
    // influx_url: url, // use '/influx' proxy to avoid problem with InfluxDB v2 Beta (Docker)
    influx_token: token,
    influx_org: org,
    influx_bucket: bucket,
    id,
  } = config
  const queryApi = new InfluxDB({url: '/influx', token}).getQueryApi(org)
  const result = await queryTable(
    queryApi,
    flux`
  import "influxdata/influxdb/v1"
  from(bucket: ${bucket})
    |> range(start: ${fluxDuration(timeStart)})
    |> filter(fn: (r) => r._measurement == "environment")
    |> filter(fn: (r) => r["_field"] == "Temperature" or r["_field"] == "TVOC" or r["_field"] == "Pressure" or r["_field"] == "Humidity" or
r["_field"] == "CO2" or r["_field"] == "Lat" or r["_field"] == "Lon")
    |> filter(fn: (r) => r.clientId == ${id})
    |> v1.fieldsAsCols()`
  )
  return result
}
```

Flux Geo Functions

Filtering

- filterRows()
- gridFilter()
- strictFilter()

Aggregate

- groupByArea()
- asTracks()

Transformation

- s2CellIDToken()
- toRows()

Supported Shapes

- **box** - defined by: minLat, maxLat, minLon, maxLon
- **circle** - defined by: lat, lon, radius
- **polygon** - array of points: lat, lon

```
import "experimental/geo"

//Circle
from(bucket: "rides")
  |> range(start: 2019-11-01T00:00:00Z)
  |> filter(fn: (r) => r._measurement == "bike")
  |> geo.filterRows(region: {lat: 40.69335938, lon: -73.30078125, radius: 20.0})

//Box
from(bucket: "rides")
  |> range(start: 2019-11-01T00:00:00Z)
  |> filter(fn: (r) => r._measurement == "bike")
  |> geo.filterRows(region: {minLat: 40.51757813, maxLat: 40.86914063, minLon: -73.65234375, maxLon: -72.94921875})

//Polygon
from(bucket: "rides")
  |> range(start: 2019-11-01T00:00:00Z)
  |> filter(fn: (r) => r._measurement == "bike")
  |> geo.filterRows(region: {points:[{lat: 40.671659, lon: -73.936631}, {lat: 40.706543, lon: -73.749177},{lat: 40.791333, lon: -73.880327}]})

// Filter if GEO hashtag is not available - slow
from(bucket: "rides")
  |> range(start: 2019-11-01T00:00:00Z)
  |> filter(fn: (r) => r._measurement == "bike")
  |> geo.toRows()
  |> geo.strictFilter(region: {minLat: 40.51757813, maxLat: 40.86914063, minLon: -73.65234375, maxLon: -72.94921875})

// The fastest GEO filtering - approximate results
from(bucket: "rides")
  |> range(start: 2019-11-01T00:00:00Z)
  |> filter(fn: (r) => r._measurement == "bike")
  |> geo.gridFilter(region: {minLat: 40.51757813, maxLat: 40.86914063, minLon: -73.65234375, maxLon: -72.94921875})
  |> geo.toRows(correlationKey: ["_time", "id"])
  |> geo.asTracks()
```

2.1 Leaflet library imports

Use **Leaflet** library

Add the highlighted imports

```
import {colorLink, colorPrimary, colorText} from'../styles/colors'
import {IconRefresh, IconSettings} from'../styles/icons'


import {MapContainer, TileLayer} from 'react-leaflet'
import AntPathWrapper from '../util/antPathWrapper'

interface DeviceConfig {
```

2.2 Create Visualisation component

Add the following function before **renderPlot()**

```
const geo =
  measurementsTable && measurementsTable?.length
  ? (() => {
    const latCol = measurementsTable.getColumn( 'Lat', 'number') as number[]
    const lonCol = measurementsTable.getColumn( 'Lon', 'number') as number[]
    const last = <T,>(arr: T[]) => arr[arr.length - 1]
    if (!lonCol || !latCol) return undefined

    const track = latCol.map<[number, number]>((x, i) => [x, lonCol[i]])

    // Made from basic react-leaflet example https://react-leaflet.js.org/docs/start-setup
    return (
      <>
        <MapContainer
          style={{width: '100%', height: '500px'}}
          center={track.length ? track[track.length - 1] : undefined}
          zoom={6}
        >
          <TileLayer
            attribution='&copy; <a href="http://osm.org/copyright">OpenStreetMap</a> contributors'
            url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
          />
          <AntPathWrapper positions={track} />
        </MapContainer>
        <Divider />
      </>
    )
  })()
  : undefined
```

2.3 Add GEO visualisation into dashboard

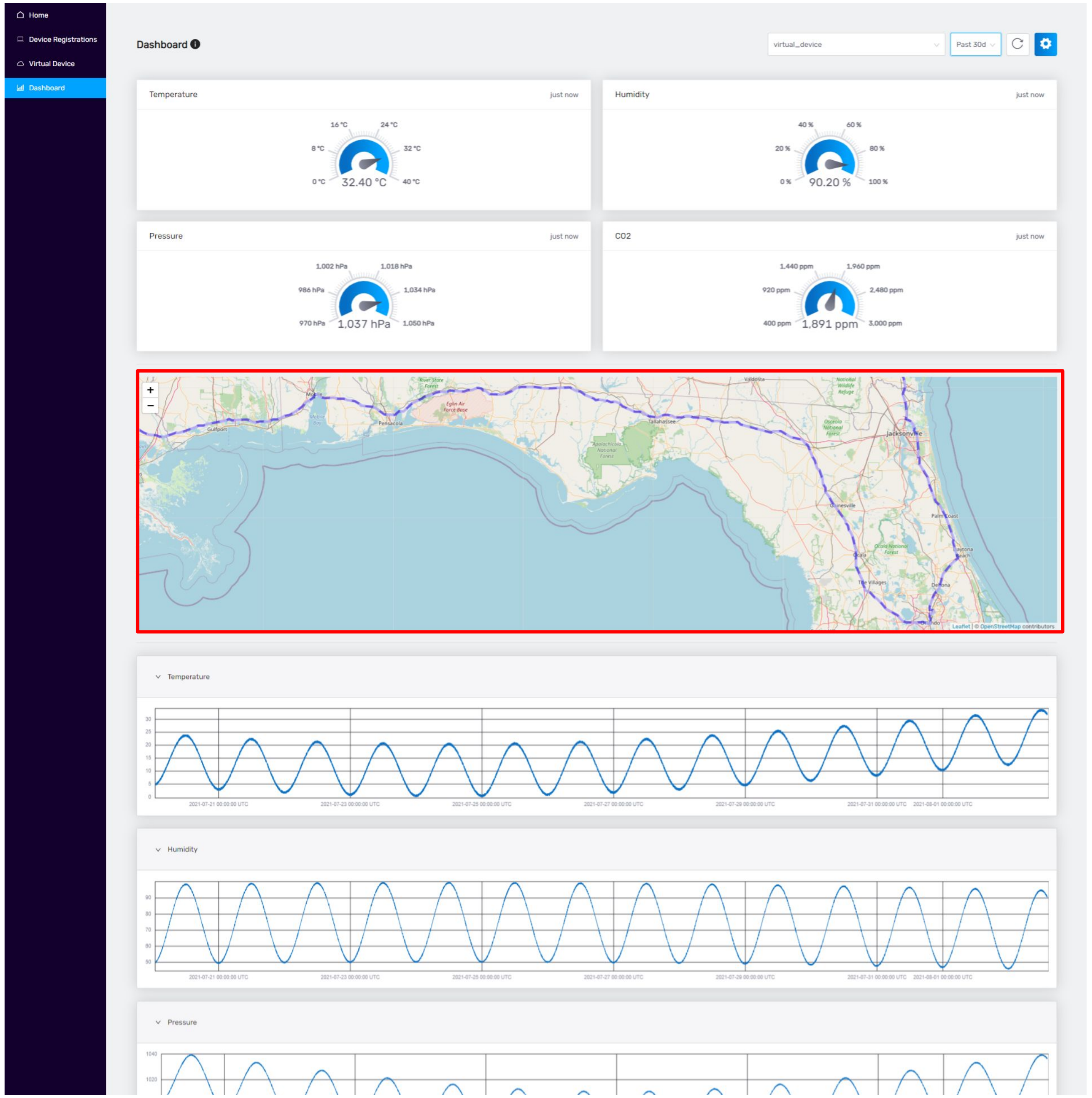
Add highlighted line adding the geo visualisation

```
      {deviceData?.measurementsTable?.length ? (
        <>
          {gauges}
          {geo}
          {plots}
        </>
      ) : (
        <Card>
          <Empty />
        </Card>
      )}
    </PageContent>
  )
}

export default DashboardPage
```

Dashboard with GEO

Refresh page if not visible



Break - 5 minutes

05:00

Does not work?

- run
  - **cd iot-center-v2/app/ui/src/pages**
  - **git checkout origin/map -- DashboardPage.tsx**
- refresh Dashboard page if geo is not visible