# Visualisation de Bases de Données sur Carte

Omar Roland Baumgärtner
Haute École Bruxelles-Brabant
omar.baumgartner@gmail.com

**Résumé**

*Cet article décrit les étapes réalisées afin de développer un programme en langage Python, capable de traiter plusieurs types de données issues de bases de données différentes afin de les visualiser sur une même carte géographique.*

## Mots-clés

Python, Visualisation, Base de données, Carte, Folium

## I. Introduction

L'objectif de ce projet consiste à développer une application en langage Python permettant le traitement de différentes bases de données afin de les visualiser sur carte géographique. Le projet se base intégralement sur la partie informatique qui commence par la recherche et le traitement de bases de données, en passant par l'implémentation d'une interface utilisateur jusqu'à la visualisation des données pertinentes sur carte.
Ce type d'application peut, dans tous les domaines, servir à analyser des données structurées brutes qui ne peuvent normalement pas être traitée manuellement par l'Homme (vu leurs quantités massives), par faute de temps et/ou de moyens humain, et qui permet donc de faciliter l'analyse et l'établissement de corrélations entre les données.

## II. Bibliothèques & Méthodes

L'application doit permettre à l'utilisateur, à travers une interface graphique, de sélectionner les facteurs à représenter, et cela sur une période choisie.
Puis, lors de la confirmation du choix de l'utilisateur, le programme accède aux bases de données concernées et récupère les données comprises dans l'intervalle de période sélectionnée afin de les convertir dans un format capable d'être lu par la bibliothèque chargée de générer la carte sous format HTML.
Le développement du projet s'est donc fait en plusieurs étapes, en commençant d'abord par la recherche des bases de données, puis par le développement de l'interface graphique, et enfin du développement du programme s'occupant des traitement des données et de la visualisation de ces dernières sur carte.

Voici le schéma fonctionnel de l'application permettant de comprendre son fonctionnement :

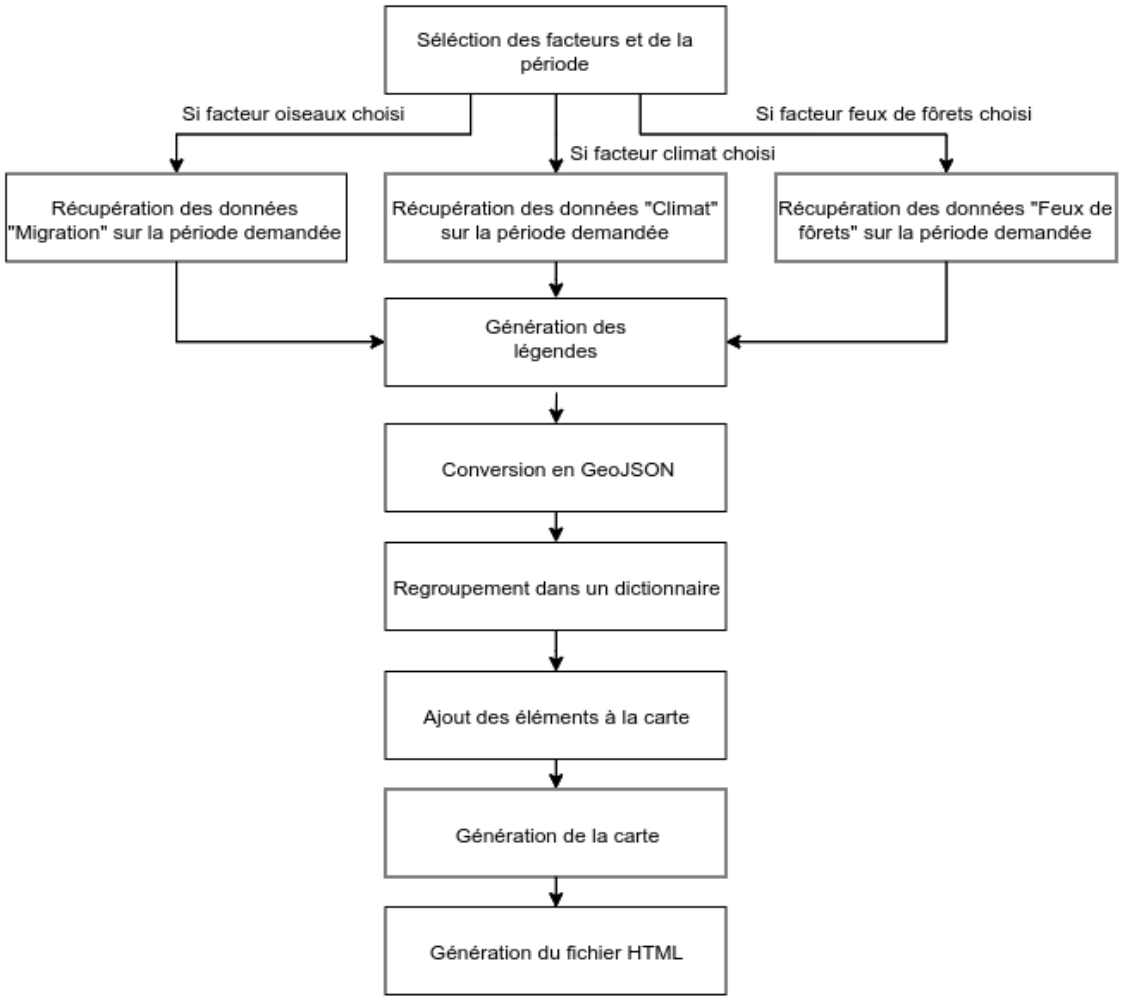


**Figure 1 –** *Schéma fonctionnel de l'application*

Notons que la recherche de bases de données satisfaisant les critères de cohérence, de pertinence et de fiabilité est crucial. Dans notre cas, les bases de données choisies sont celles qui concernent : les feux de forêts en Afrique, les températures de surface de plusieurs pays, et la migration des goélands marins.

- Migrations des oiseaux
  Les données recueillies sont sous format CSV et proviennent de MoveBank[1], une base de données de suivi d'animaux supervisés par l'iInstitut Max-Planck d'Ornithologie en Haute-Bavière, Allemagne, et plus précisément de l'ornithologue allemand Martin Wikelski. Ces données regroupent le suivi de 33 Goélands marin sur une période de 7 ans de 2009 à 2015, et contiennent trois attributs qui nous seront utiles : L'identifiant de l'oiseau, sa position (longitude, latitude) et la date correspondante.

- Température de surface des pays
  Pour cette partie, nous avons eu besoin de deux bases de données.
  La première concerne les températures de surface de chaque pays de 1743 à 2013 et provient de Data.world[2], la plus grande communauté de données collaborative au monde.
  La deuxième base de données provient de Natural Earth[3], qui contient un ensemble de données cartographiques du domaine public ( sous format DBF ), et qui regroupe 247 pays et leurs surfaces sous forme de polygones de coordonnées afin de pouvoir les représenter sur carte.

- Feux de forêts
  Ces données ont été récupérées de la NASA[4], et plus précisément du FIRMS (Fire Information for Resource Management System) qui contient les informations sur les incendies pour le système de gestion des ressources, fournies sous format DBF. La Nasa donne donc accès à des données sur les incendies actifs en temps quasi réel ou archivés. Elle obtient ces données par radiomètre spectral pour imagerie de résolution moyenne (MODIS, Moderate-Resolution Imaging Spectroradiometer), ou par radiomètre pour imagerie à infrarouge visible (VIIRS, Visible Infrared Imaging Radiometer Suite).
  Dans notre cas, les données proviennent de la méthode MODIS et comptent plus de 14 millions d'enregistrements de feux de forêts en Afrique de 2010 à 2013.

Pour que toutes ces données soient en concordance, il faudra donc prendre une période comprise entre 2010 et 2013. Pour des raisons de performances, les bases de données ont été nettoyées de telle sorte que les données dépassant l'intervalle étudié sont supprimés afin que le programme prenne moins de temps à les parcourir.
Nous avons ensuite développée l'interface graphique, qui va permettre à l'utilisateur de sélectionner la période souhaitée ( comprise entre 2010 et 2013 ), et les facteurs à afficher. Nous avons utilisé la bibliothèque Tkinter, qui est la bibliothèque graphique libre d'origine pour le langage Python pour créer une interface graphique.
Pour la partie traitement des données, les bibliothèques Pandas et GeoPandas nous ont permis de parcourir les fichiers CSV et les bases de données de format DBF contenant des données classiques ( texte brut ) ou géospatiales ( format SHP, ShapeFile ), qui devront ensuite être converties et retournées sous format GeoJSON. Rappelons que le format GeoJSON est une structure de données semblable à :

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

**Figure 2 –** *Format GeoJSON*

Chaque donnée est considérée comme étant un “feature”, caractérisé par sa géométrie ( point, ligne, polygone..), ses coordonnées et ses propriétés.
Les features sont ensuite regroupées dans un même dictionnaire qui, contrairement à la structure ordonnée d’une liste utilisant des indices, utilise des clés pour accéder aux objets. Ce dictionnaire fera guise de FeatureGroup, qui sera récupéré par la bibliothèque Folium afin de générer une carte géographique sous format HTML.
Folium[5] est une interface entre python et la bibliothèque Leaflet (- qui est une bibliothèque JavaScript ) qui se base sur les données cartographique d’OpenStreetMap pour générer des cartes dynamiques, et utilise Bootstrap pour la représentation des données ( design des icônes ) ce qui en fait son avantage au niveau esthétique et surtout ergonomique. Il permet par ailleurs l’ajout de fonctionnalités tel que le contrôle du niveau de grossissement, l’ajout d’un champ indiquant l’échelle de grossissement et l’ajout d’un champ montrant les coordonnées de la position de la souris.
Nous avons par ailleurs utilisé la bibliothèque Pillow qui est la bibliothèque d’imagerie Python qui nous a permis de générer des images qui feront guise de légendes.

## III. Résultats

La figure ci-dessous représente l’interface graphique qui permettra de générer la carte.

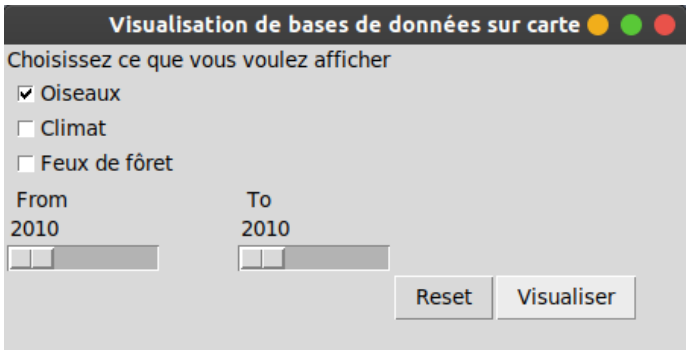


**Figure 3 –** *Interface de l’application*

Après l’ouverture automatique du fichier HTML contenant la carte géographique, nous pouvons voir en premier lieu un curseur de temps permettant de sélectionner une date précise, ou de faire défiler les dates à une vitesse choisie. Une échelle de grossissement est située en dessous du TimeSlider et permet d’avoir une idée sur le zoom que l’on fait.

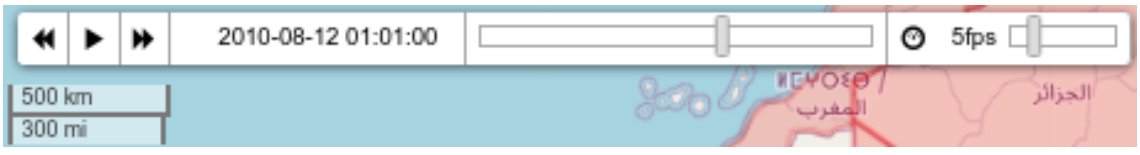


**Figure 4 –** *Curseur temporel et échelle de grossissement*

Pour la partie concernant la température de surface des pays, nous avons comme légende une échelle de température qui varie de -30 à 40 degrés, allant du bleu au rouge afin de colorer les pays.

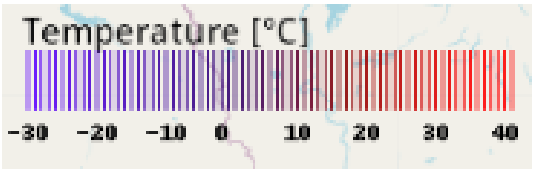


**Figure 5 –** *Nuancier de couleurs pour la partie climat*

Le résultat obtenu pour cette partie est le suivant :

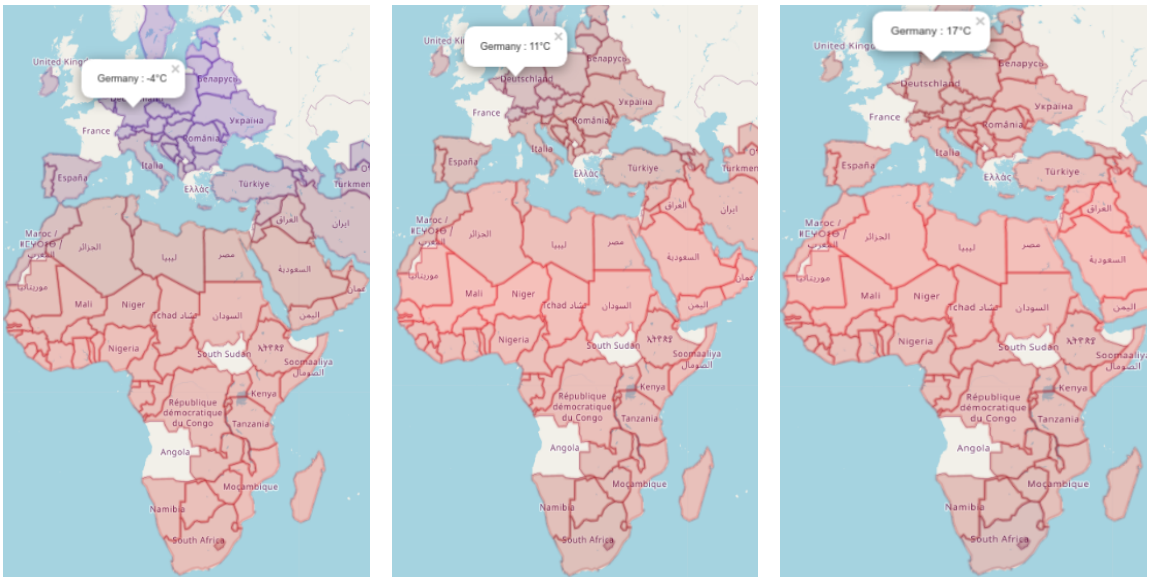


**Figure 6 –** *Affichage des données relatives aux températures de surface des pays*

Pour la partie "Migration des oiseaux", chaque compilation génère des couleurs aléatoires qui sont attribués aux oiseaux et une légende est par ailleurs générée à partir de ses couleurs et de leurs identifiants afin de pouvoir localiser chaque oiseau.


**Figure 7 –** *Légende regroupant les oiseaux*

Le résultat obtenu pour cette partie est le suivant :

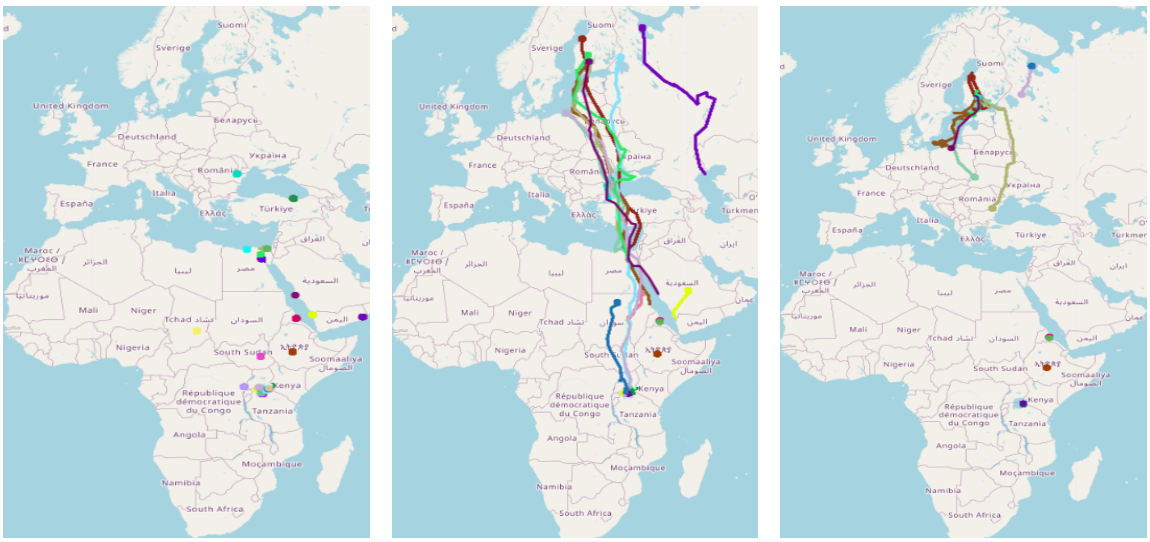

**Figure 8 –** *Affichage des données relatives à la migration des oiseaux*

Pour le dernier facteur, les feux de forêts sont représentés par des points rouges comme représenté ci-dessous :

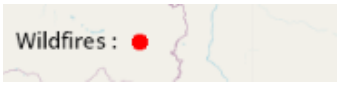


**Figure 9 –** *Légende représentant les feux de forêts*

Le résultat obtenu pour cette partie est le suivant :

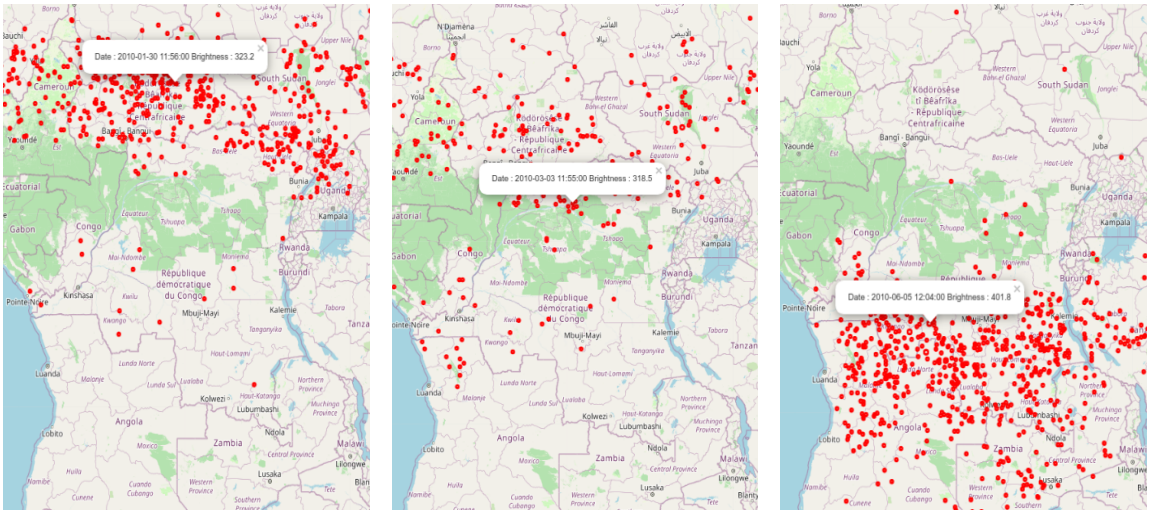


**Figure 10 –** *Affichage des données relatives aux feux de forêts*

En combinant les 3 fonctionnalités, on obtient le résultat suivant :

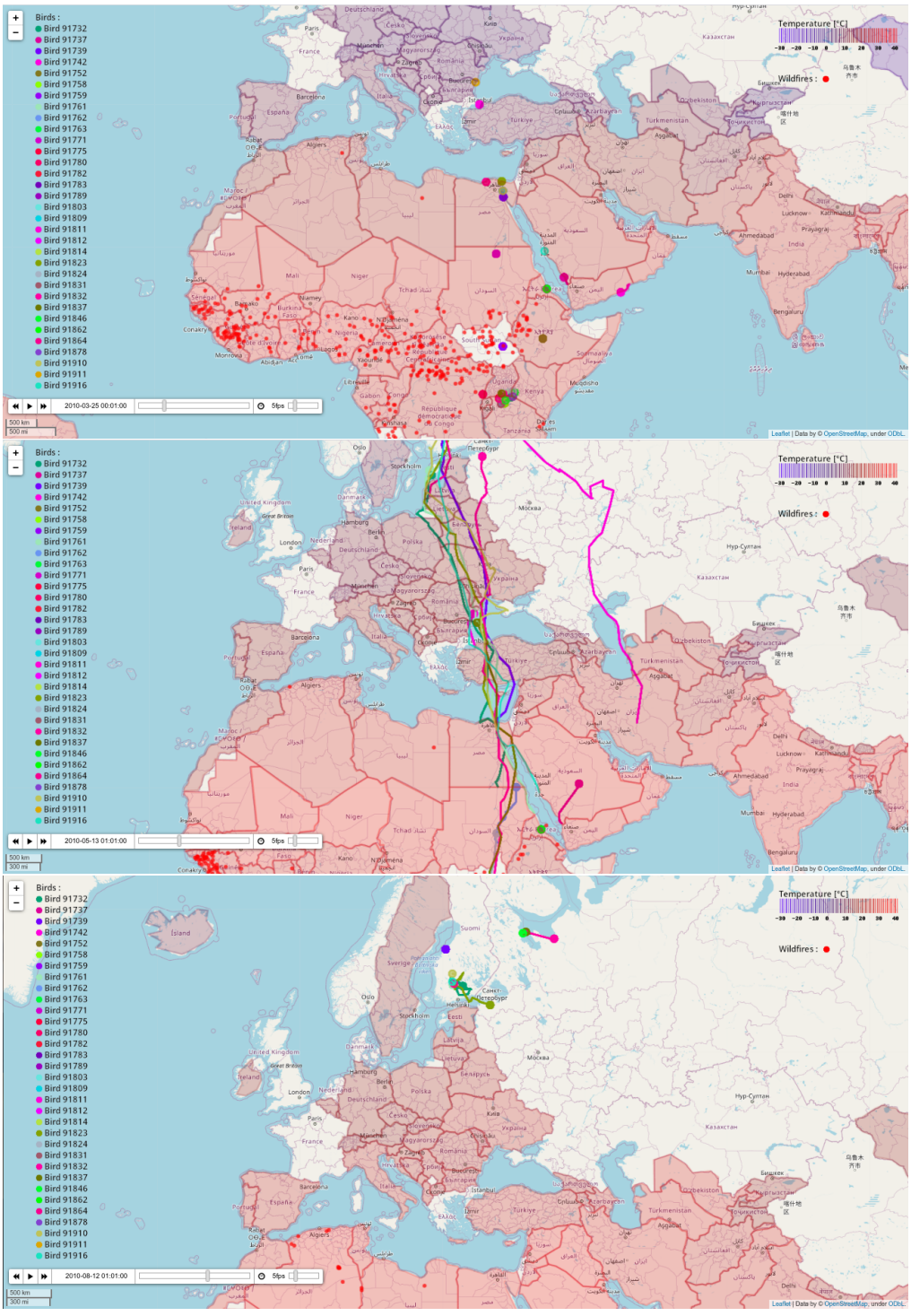


**Figure 11 –** *Produit final*

## IV. Conclusions et améliorations possibles

Pour conclure, nous pouvons avancer que plusieurs problèmes ont été rencontrés que ce soit avant et pendant le développement.
La difficulté avant le développement a été la recherche et l'identification de bases de données gratuites, complètes, de sources fiables, et cela tout en restant sur une même période.
Un autre problème majeur rencontré pendant le développement a concerné la bibliothèque Folium : certaines méthodes manquent de documentations, tandis que d'autres méthodes ne sont pas encore complètement développées et ne sont donc pas encore fonctionnelles, ce qui a donc fortement ralenti le développement du projet.
Cela dit, nous pouvons avancer que l'objectif de visualisation de données sur carte est atteint de manière satisfaisante. Ce projet fût un très bon exercice d'application qui m'a permis d'apprendre comment traiter des données massives.
En outre, cette même technologie peut être utilisée dans d'autres domaines tel que dans le domaine de la santé (épidémiologie), en prenant l'exemple de la représentation sur carte des cas de COVID-19 recensés dans le monde et voir les corrélations avec la densité de la population, le niveau de vie, etc.. Ou encore dans le domaine économique pour identifier les corrélations entre le niveau de développement d'un pays, le PIB, le taux de chômage, les investissements en R&D...
Chaque travail peut toujours être amélioré ou rendu plus puissant. Dans notre cas, et dans le cadre de ce travail, nous pouvons proposer de faire évoluer l'application de manière à la rendre totalement polyvalente en permettant à l'utilisateur d'insérer ses propres bases de données et les attributs pertinents qui l'intéressent, afin de donner à l'application une infinité d'utilisations possibles.

## Références

[1] MoveBank, True navigation in migrating gulls

[2] Data.world, Global Climate Change Data

[3] Natural Earth Data, 1:110m Cultural Vectors

[4] Nasa, FIRMS Archive Download

[5] Documentation Folium