

# Projet

*Dashboard GéoAnalytique à l'aide de Streamlit*  
*Web mapping*

*Réalisé par :*

ABBOUBI Zaineb N°1

EL FETTAHI Fadwa N°22

*Encadré par :*

*Prof. Hajji Hicham*

## Table de matière :

Introduction : .....	3
I. Construction du Dataset : .....	3
II. Utilisation de Streamlit Géospatial :.....	6
a) Volet cartographie :.....	6
b) Slider :.....	8
c) Timelapses :.....	10
d) SplitMap.....	12
e) Pop-Up des séries temporelles sous forme de graphique :.....	14
f) Recherche des points par ses coordonnées :.....	17
g) Requêtes spatiales : .....	19
h) Exploration de l'utilisation du format COG pour les cartes Raster : .....	21
i) Export du contenu du Dashboard :.....	23

## *Introduction :*

Le présent rapport synthétise l'ensemble du processus de création un Dashboard GeoAnalytique en utilisant la librairie streamlit.

Ce projet vise à développer nos compétences en web mapping tout en explorant les fonctionnalités de Streamlit pour créer une plateforme interactive facilitant l'exploration de données géographiques, et offrant ainsi une valeur ajoutée dans le contexte de l'analyse spatio-temporelle.

## *I. Construction du Dataset :*

Pour générer le jeu de données nécessaire à notre Dashboard GeoAnalytique, nous avons développé un script en langage Python. Ce script utilise la bibliothèque GeoPandas pour charger un fichier GeoJSON délimitant la frontière du Maroc. Ensuite, il génère aléatoirement des coordonnées de 2000 points à l'intérieur de cette frontière et procède à la création de données en respectant le schéma de données spatio-temporelles suivant :

<b>Geométrie</b> : point	Attibut2Jour-3 : numérique (entre 0 et 20)
Propriete1 : texte	Attibut2Jour-4 : numérique (entre 0 et 20)
Propriete2 : numérique relatif	Attibut2Jour-5 : numérique (entre 0 et 20)
Propriete3 : numérique relatif	Attibut2Jour-6 : numérique (entre 0 et 20)
Propriete4 : Date	Attibut3Jour0 : numérique (entre 0 et 50)
Attibut1Jour0 : numérique (entre 0 et 100)	Attibut3Jour-1 : numérique (entre 0 et 50)
Attibut1Jour-1 : numérique (entre 0 et 100)	Attibut3Jour-2 : numérique (entre 0 et 50)
Attibut1Jour-2 : numérique (entre 0 et 100)	Attibut3Jour-3 : numérique (entre 0 et 50)
Attibut1Jour-3 : numérique (entre 0 et 100)	Attibut3Jour-4 : numérique (entre 0 et 50)
Attibut1Jour-4 : numérique (entre 0 et 100)	Attibut3Jour-5 : numérique (entre 0 et 50)
Attibut1Jour-5 : numérique (entre 0 et 100)	Attibut3Jour-6 : numérique (entre 0 et 50)
Attibut1Jour-6 : numérique (entre 0 et 100)	
Attibut2Jour0 : numérique (entre 0 et 20)	
Attibut2Jour-1 : numérique (entre 0 et 20)	
Attibut2Jour-2 : numérique (entre 0 et 20)	

Figure 1: schéma de données spatio-temporel.

Le jeu de données ainsi généré est ensuite enregistré au format GeoParquet,

## ➔ Nature des données et possibilités de cartographie :

Le choix de nos données générées repose sur une sélection diversifiée d'événements spatio-temporels, chacun étant caractérisé par une multitude de paramètres. Parmi ces paramètres, on retrouve le type d'événement (Propriété1), qui englobe des catégories telles que l'Incendie de forêt, la Chute de neige, l'Inondation et l'Accident de la route. Également pris en considération, la gravité associée à chaque événement (Propriété2), la variation temporelle (Propriété3), la date de l'événement, ainsi que des données météorologiques comprenant l'humidité, les précipitations et la température pour la date de l'événement et les six jours précédents.

	Type d'Événement	Gravité de l'événement ▾	Variation Temporelle	Date(jour0)	humidityJour0(%)	humidityJour-1(%)	humidityJour-2(%)	humidityJour-3(%)	humidityJour-4(%)
309	Accident de la route	42.29710390988162	10.424686040693413	2023-09-01	4.91	37.89434891974253	47.41676252815471	37.14554904950954	15.7703530916...
310	Accident de la route	42.29609938814827	-18.764451118931834	2023-08-01	66.57	7.434291106471836	43.43806669629847	16.316452354035725	25.0207777446...
311	Accident de la route	42.28980708170073	-19.989597153065645	2023-08-01	30.48	17.522751947847958	48.76922912081928	88.03947353077959	11.9889320825...
312	Inondation	42.255200037659975	11.465690474357451	2023-09-01	33.01	87.21609180062899	88.00331460048831	42.362998210337466	86.6995692167...

	humidityJour-5(%)	humidityJour-6(%)	precipitationJour0(cm/m²)	precipitationJour-1(cm/m²) ▲	precipitationJour-2(cm/m²)	precipitationJour-3(cm/m²)
309	29.188553809440364	32.48586077762183	1.02	3.0554106738166187	1.8767477239834873	10.639461154418996
310	35.095598671409064	1.108563277441088	17.7	3.0569498393610295	15.700907358491737	11.563946055709817
311	24.323400489970126	50.75762994880145	5.95	3.061211918036759	10.427030226697955	12.08939605134632
312	78.58771624707272	18.84099517164285	19.09	3.06314790998238	12.279679923292399	7.108668614226358

	precipitationJour-5(cm/m²)	precipitationJour-6(cm/m²)	temperatureJour0(°C)	temperatureJour-1(°C)	temperatureJour-2(°C)	peratureJour-3	peratureJour-4	nperatureJour-5(°	peratureJour-6
309	6.253593789199108	0.509013444144093	24.76	34.500557983478366	36.14831485504439	10.1044997...	2.18145974...	7.74700726596827	11.3200110...
310	10.427430600079147	7.7025268851452005	21.37	41.99379954812175	16.368169156185125	47.5853755...	37.4372385...	7.78525256098252	33.4232933...
311	9.25496577948709	13.523050400266639	39.57	3.35201507791133	9.286840902478454	47.1654244...	15.0690043...	7.819863604822...	34.5776111...
312	19.504180219696977	15.997232828962735	46.99	37.369467813715964	4.094650162051305	39.6918617...	9.09458445...	7.824059925047...	33.5476216...

Figure 2 : Aperçus de notre donnée.

## A) Discussion sur l'utilité du format GeoParquet :

Le choix du format de stockage des données revêt une importance capitale dans la conception de notre Dashboard GeoAnalytique. La question qui se pose naturellement est la suivante : quelle est l'utilité du format GeoParquet dans notre contexte spatio-temporel spécifique ?

Pour répondre à cette interrogation, nous avons sauvegarder nos données générées dans divers formats en vue d'une comparaison. Le tableau ci-dessous présente un résumé des avantages et des inconvénients de chaque format de stockage par rapport au GeoParquet.








Nom	Taille
 data.geojson	2 352 Ko
 data.gpkg	768 Ko
 data.dbf	1 394 Ko
 data.shp	55 Ko
 data.shx	16 Ko
 data.cpg	1 Ko
 DATA.parquet	454 Ko

Figure 3 : Différent formats de stockage de Nos données et leurs tailles.

GeoparQuet	Shp
<ul style="list-style-type: none"> <li>- Format de stockage de données en colonnes</li> <li>- Compression efficace</li> <li>- Prise en charge du partitionnement des données</li> <li>- Permet le Traitement parallèle</li> <li>- bien intégré à Apache Arrow (projet open source de la Fondation Apache), ce qui peut faciliter l'interopérabilité et l'intégration avec d'autres outils de traitement de données.</li> </ul>	<ul style="list-style-type: none"> <li>- Grand taille de stockage</li> <li>format multi-fichiers (.shp, .shx, .dbf et .prj )</li> <li>- Limité par la taille de son nom de colonne (10 caractères)</li> <li>- A une limite de taille de 2 Go</li> </ul>
	GeoJson
	<ul style="list-style-type: none"> <li>- N'est pas compressé</li> <li>- Stocké sous forme de texte brut, donc sa taille est énorme.</li> </ul>
	GeoPackage
	-Taille plus grande que GeoParquet

### Dans notre cas :

Afin de mettre en valeur les avantages offerts par GeoParquet, Dans le contexte de la question (e), nous avons réalisé un parallélisme et une partition de nos données, permettant ainsi d'améliorer l'efficacité de notre code et d'accélérer son temps d'exécution.

```
# Diviser le GeoDataFrame en partitions
num_partitions = 4
gdf_partitions = np.array_split(filtered_gdf, num_partitions)

# Traitement parallèle des partitions
with ThreadPoolExecutor() as executor:
    all_markers = list(executor.map(process_partition, gdf_partitions))
```

Figure 4 : partie du code de la question : e.

## *II. Utilisation de Streamlit Géospatial :*

### **a) Volet cartographie :**

Pour cette partie, nous avons développé un code permettant aux utilisateurs d'explorer de manière interactive nos données géospatiales. Il offre la possibilité de visualiser la distribution spatiale ainsi que la variation des différentes propriétés des événements cartographiés en fonction de la date choisie.

#### **➔ Description rapide du code :**

##### **1. Importation des Bibliothèques :**

- Importation des bibliothèques essentielles, notamment Streamlit, GeoPandas, et Plotly Express.

##### **2. Chargement des Données :**

- Chargement des données au format GeoParquet à l'aide de GeoPandas.

##### **3. Sélection de la Date et de la Colonne :**

- Utilisation de widgets Streamlit pour permettre à l'utilisateur de choisir une date spécifique (jour0) et la colonne à cartographier parmi plusieurs options.

##### **4. Filtrage des Données :**

- Filtrage du GeoDataFrame en fonction de la date sélectionnée.

##### **5. Création de la Carte avec Plotly Express :**

- Utilisation de Plotly Express pour générer une carte thématique basée sur la colonne sélectionnée. Personnalisation de la carte avec des informations telles que le titre et le style.

##### **6. Affichage avec Streamlit :**

- Affichage de la carte générée dans l'interface Streamlit à l'aide de la fonction `st.plotly_chart`.

##### **7. Gestion des Données Absentes :**

- Affichage d'un avertissement si aucune donnée n'est disponible pour la date sélectionnée.

Résultat de l'application :

# Bienvenue dans le Dashboard GeoAnalytique!

Explorez les données géographiques en sélectionnant une date et une colonne à cartographier. Utilisez les options ci-dessous pour personnaliser votre expérience.

Choisissez la date (jour0)

2023-07-01

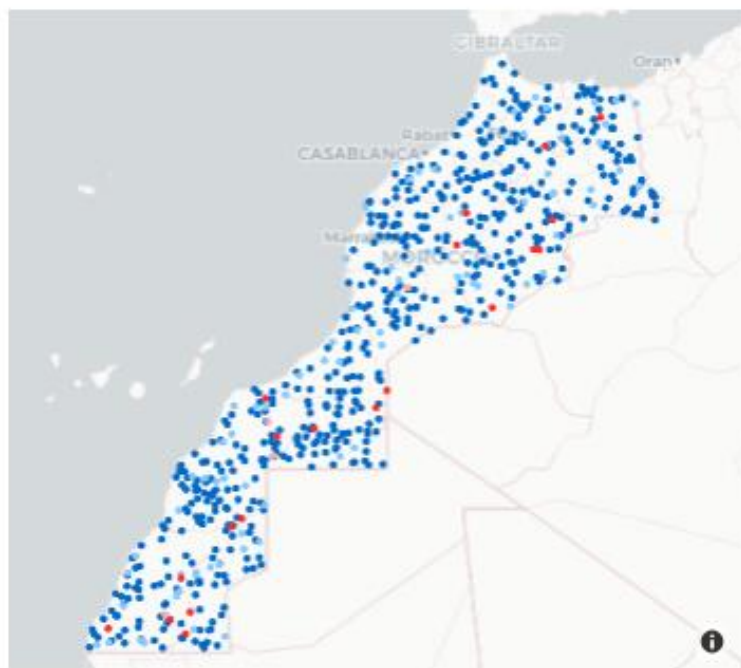


Choisissez la colonne à cartographier

Type d'Événement



## Carte - Type d'Événement (2023-07-01)



Type d'Événement

- Accident de la route
- Inondation
- Chute de neige importante
- Incendie de forêt

Pour la carte du type d'événement, double cliquez sur la légende pour isoler une trace.



## b) Slider :

Dans cette partie on présente une application web permettant d'explorer des cartes météorologiques basées sur des données interpolées (cartes Raster). Pour visualiser ces cartes, nous avons créé 63 cartes sur QGIS en utilisant la méthode IDW pour effectuer une interpolation. L'utilisateur peut sélectionner une date, un phénomène météorologique (humidité, précipitations, température) et naviguer entre différents jours à l'aide d'un curseur (slider). Les cartes sont générées en correspondance à la date et au phénomène choisis.

### ➔ Description rapide du code :

#### 1. Chargement des bibliothèques :

- Le code commence par importer les bibliothèques nécessaires, notamment Streamlit, Leafmap et d'autres modules associés.

#### 2. Fonctions de Chargement et d'Affichage des Cartes :

- **load\_map(date, phenomenon, day\_offset)**: Cette fonction génère le chemin du fichier COG en fonction de la date, du phénomène météorologique et de l'offset du jour. Elle vérifie ensuite si le fichier existe et le renvoie.
- La fonction principale **main()** utilise Streamlit pour créer l'interface utilisateur. Elle propose des options de sélection pour la date, le phénomène et l'offset du jour.

#### 3. Création de la Carte Leafmap :

- Une carte Leafmap est créée avec un centre sur le Maroc et un zoom ajusté.
- L'image COG est ajoutée à la carte avec la fonction **add\_raster()**.
- Une légende colorée est ajoutée en fonction du phénomène météorologique choisi avec **add\_colorbar()**.

#### 4. Affichage avec Streamlit :

- L'interface utilisateur, créée avec Streamlit, propose des menus déroulants et un curseur pour l'interaction.
- La carte générée avec Leafmap est ensuite affichée dans l'interface Streamlit à l'aide de **folium\_static()**.



Résultat de l'application :

# Application de Cartes Météo

Explorez les variations spatiales des événements météorologiques tels que l'humidité, la température et les précipitations à travers le temps. Utilisez les options de sélection pour choisir la date, le phénomène météorologique et naviguez entre les différents jours. Les cartes sont générées à partir de données interpolées, offrant ainsi une visualisation dynamique et informative.

Sélectionner la date

2023-07-01

Sélectionner le phénomène

humidity

Sélectionner le jour (aujourd'hui = 0, hier = -1, etc.)



## c) Timelapses :

Pour réaliser les timelapses, nous avons utilisé un script qui simplifie l'exploration de timelapses interactifs, permettant une visualisation dynamique des conditions météorologiques : la température, la précipitation et l'humidité, sur une période de 6 jours avant la date sélectionnée.

- Ces visualisations sont rendues possibles grâce à nos cartes interpolées créées sur QGIS.

### ➡ Description rapide du code :

#### 1. Importation des Bibliothèques :

- Importation des bibliothèques nécessaires, telles que Streamlit, base64, et leafmap, pour assurer la fonctionnalité du code.

#### 2. Titre et Introduction :

- Utilisation de **st.markdown** pour afficher un titre captivant et une introduction détaillée, offrant une perspective claire sur l'objectif du code.

#### 3. Sélection de la Date et de l'Attribut :

- Intégration de widgets Streamlit (st.selectbox) pour permettre à l'utilisateur de choisir la date et le type de données météorologiques à visualiser à travers les timelapses.

#### 4. Construction du Chemin des Images :

- Construction du chemin des images en fonction de la sélection de l'utilisateur

#### 5. Génération du Timelapse avec Leafmap :

- Utilisation de la bibliothèque Leafmap pour créer des timelapses interactifs. L'appel de la fonction **leafmap.create\_timelapse** avec des paramètres spécifiques contribue à la création d'un rendu visuel dynamique et informatif.

#### 6. Affichage du Timelapse :

- Affichage du timelapse généré dans l'interface Streamlit, offrant aux utilisateurs une expérience visuelle immersive des conditions météorologiques sur une période de 6 jours avant la date choisie.

Résultat de l'application :

## Analyse des Données Météorologiques par les timelapses.

Explorez des timelapses interactifs offrant une visualisation dynamique des conditions météorologiques sur une période de 7 jours. Nos timelapses utilisent des cartes interpolées basées sur la méthode IDW pour fournir une représentation spatiale lissée des données météorologiques au Maroc.

Sélectionner la date

2023-08-01

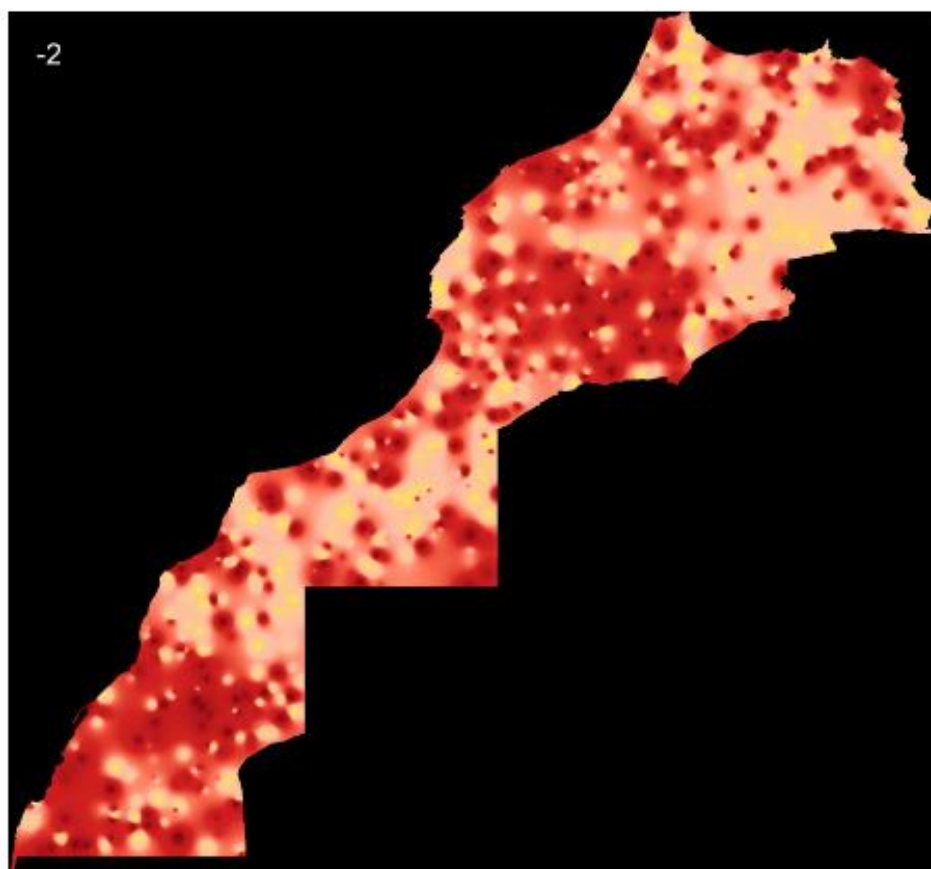


Choisissez le type de données à visualiser

temperature



Le timelapse correspondant :



## d) SplitMap

Pour cette partie on vise à créer une application web dynamique permettant de comparer de manière évolutive des cartes météorologiques pour deux jours consécutifs. L'utilisateur peut sélectionner une date et un phénomène météorologique spécifiques, puis utiliser un curseur pour naviguer entre les cartes du jour actuel et du jour précédent. L'application utilise la bibliothèque **Leafmap** et **folium** pour intégrer des cartes interactives avec Streamlit.

### → Description rapide du code :

#### 1. Création de la Carte Leafmap :

Le code commence par créer une carte Leafmap avec un centre défini sur les coordonnées [39.4948, -108.5492] et un zoom de niveau 12.

#### 2. Chargement des Images de Carte :

La fonction **load\_map(date, phenomene, day\_offset)** génère les chemins des fichiers COG en fonction de la date, du phénomène météorologique et de l'offset du jour. Elle vérifie ensuite si les fichiers existent.

#### 3. Interface Utilisateur Streamlit :

- L'interface utilisateur est créée avec Streamlit, présentant un titre, une introduction et des menus déroulants pour la sélection de la date et du phénomène.
- Un slider est inclus pour permettre à l'utilisateur de naviguer entre les cartes des jours actuel et précédent.

#### 4. Ajout des Images à la Carte :

- Les images des jours sélectionné et précédent sont ajoutées à la carte Leafmap avec la fonction **add\_raster()**.
- La fonction **split\_map()** de Leafmap permet de superposer les deux images et de les diviser pour une comparaison visuelle.

#### 5. Ajout de Couches Supplémentaires :

- Une couche de carte satellite (Mapbox Satellite) est ajoutée pour une référence visuelle supplémentaire.
- La position de la souris est affichée en bas à droite de la carte.

#### 6. Contrôles Interactifs :

Le code inclut un contrôle interactif de commutation de couche pour permettre à l'utilisateur de choisir entre la carte de terrain et la carte satellite.

#### 7. Affichage avec Streamlit :

Enfin, l'application est affichée avec **folium\_static()** de Streamlit.

Résultat de l'application :



## e) Pop-Up des séries temporelles sous forme de graphique :

### ➡ Description rapide du code :

#### 1. Importation des Bibliothèques :

- Importation des bibliothèques nécessaires, telles que Streamlit, Folium, Geopandas, et Altair, pour assurer la fonctionnalité du code.

#### 2. Titre et Interface Utilisateur :

- Utilisation de Streamlit pour créer une interface utilisateur.
- Affichage d'un titre HTML stylisé pour introduire le but de l'application.

#### 3. Chargement des Données :

- Chargement d'un GeoDataFrame depuis un fichier GeoParquet ('DATA.parquet').

#### 4. Sélection de la Date :

- Utilisation d'un widget selectbox pour permettre à l'utilisateur de choisir une date (jour0).

#### 5. Filtrage des Données :

- Filtrage des données géospatiales en fonction de la date sélectionnée.

#### 6. Création de la Carte avec Folium :

- Utilisation de Folium pour créer une carte centrée sur le Maroc avec un niveau de zoom prédéfini.

#### 7. Utilisation de MarkerCluster :

- Regroupement des marqueurs pour améliorer la lisibilité de la carte.

#### 8. Utilisation de la fonction process\_partition :

- Une fonction qui prend une partition de données géospatiales et crée des marqueurs pour chaque point.
- Pour chaque point, un marqueur est créé avec une position géographique définie par les coordonnées (row['Geometry'].y et row['Geometry'].x).
- Pour chaque marqueur, **un graphique Altair interactif est généré.**
- Les données pour le graphique sont extraites du DataFrame avec des valeurs de température, précipitations, et humidité sur une période de -6 jours à jour 0.
- **Les graphiques Altair sont incorporés dans des popups Folium.**

#### 9. Division du GeoDataFrame en Partitions :

- Le GeoDataFrame (filtered\_gdf) est divisé en plusieurs partitions pour un traitement parallèle. Le nombre de partitions est défini par num\_partitions.



## 10. Traitement Parallèle des Partitions :

- Les partitions du GeoDataFrame sont traitées en parallèle à l'aide d'un ThreadPoolExecutor.
- La fonction process\_partition est appliquée à chaque partition de manière concurrente, accélérant ainsi le processus de création des marqueurs.

## 11. Création de Popups avec Graphiques Altair :

- Pour chaque point sur la carte, un popup est généré.
- Chaque popup contient un graphique interactif de la série temporelle des températures, précipitations, et humidité sur une période de -6 jours à jour 0.

## 12. Ajout de Couches Supplémentaires :

- Ajout d'une couche de carte satellite (Mapbox Satellite).
- Intégration d'une mini-carte pour une vue d'ensemble.
- Affichage de la position de la souris en temps réel.
- Ajout de contrôles pour passer en plein écran.

## 13. Affichage de la Carte dans Streamlit :

- Utilisation de la fonction **folium\_static** pour afficher la carte générée avec Folium dans l'interface Streamlit.



Ce code offre aux utilisateurs la possibilité de cliquer sur les points de la carte pour afficher une série temporelle détaillée sous forme de graphique, permettant ainsi une exploration des données météorologiques géospatiales telles que la température, la précipitation et l'humidité.

Notre objectif initial pour cette fonctionnalité était de générer des **graphes interactifs** lors du clic sur un point de la carte. Malheureusement, malgré nos multiples tentatives et l'expérimentation avec différents codes pour intégrer des graphes interactifs dans les pop-ups en utilisant des bibliothèques telles que Highchart, nous obtenions uniquement des pop-ups vides au lieu des graphes interactifs escomptés.

Donc face à la complexité de l'intégration des graphes interactifs dans les pop-ups, nous avons opté pour une approche alternative en affichant un **graphique statique simple** dans le pop-up. Et **parallèlement**, nous avons créé un **graphe interactif** en dehors de la carte, et qui réagit en fonction de l'emplacement de point choisi.

Bien que cette solution diffère de notre intention initiale, elle offre toujours une exploration informative des données géospatiales.



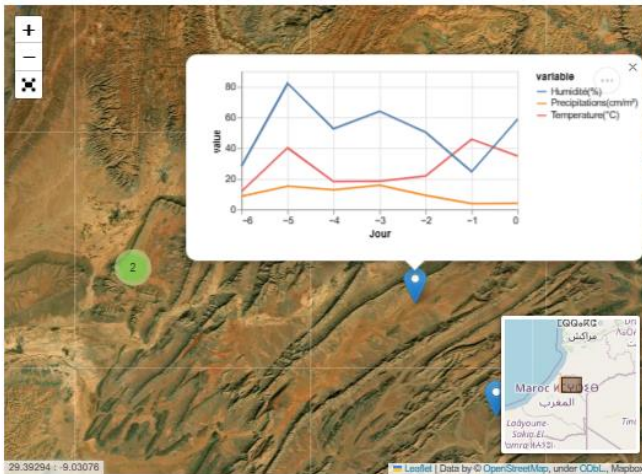
Résultat de l'application :

## Explorez la Variation des Températures, des Précipitations et de l'Humidité à Travers le Temps :

Choisissez la date (jour0)

2023-07-01

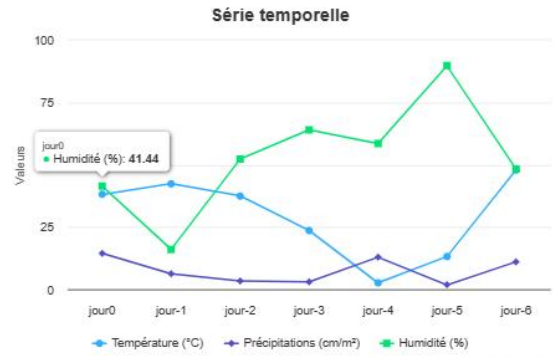
Cliquez sur un point de la carte pour afficher sa série temporelle des températures, des précipitations et de l'humidité.



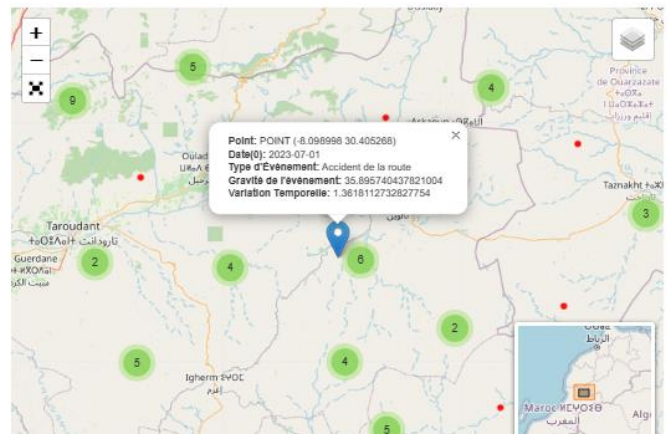
Ci-dessous, vous pouvez explorer les graphiques interactifs des séries temporelles des températures, des précipitations et de l'humidité, pour un point choisi.

Sélectionner un emplacement

POINT (-8.098998 30.405268)



Cliquez sur le marqueur de la carte pour afficher des informations détaillées sur l'emplacement choisi.



## f) Recherche des points par ses coordonnées :

On a créé une application web interactive permettant aux utilisateurs d'explorer géographiquement des points à proximité d'un emplacement spécifié. Il charge un jeu de données géographiques à partir du fichier DATA GeoParquet, permet à l'utilisateur de saisir des coordonnées (latitude, longitude), puis identifie et visualise le point dans un rayon de proximité autour de ces coordonnées. La visualisation est effectuée sur une carte interactive utilisant la bibliothèque Folium.

### ➔ Description rapide du code :

#### 1. Chargement du Fichier GeoParquet :

- Le code commence par charger un fichier GeoParquet (**DATA.parquet**) contenant des données géographiques, telles que des points avec des coordonnées et des informations associées.

#### 2. Interface Utilisateur Streamlit :

- L'interface utilisateur est créée avec Streamlit, avec un titre et une zone de saisie permettant à l'utilisateur d'entrer des coordonnées (latitude, longitude).
- Les coordonnées saisies sont ensuite utilisées pour identifier et afficher les points géographiques à proximité.

#### 3. Calcul des Points à Proximité :

- Les coordonnées saisies sont utilisées pour calculer la distance euclidienne entre ces coordonnées et chaque point dans le jeu de données.
- Les points situés dans le rayon de proximité spécifié (**proximity\_radius**) sont filtrés pour affichage.

#### 4. Affichage des Résultats sur une Carte :

- Si des points sont trouvés dans le rayon spécifié, une carte interactive est créée avec Folium.
- Les points du jeu de données sont représentés comme des cercles sur la carte.
- Le point recherché est mis en évidence avec un marqueur et un popup affichant des informations détaillées sur ce point.

#### 5. Ajout de Couches Supplémentaires :

- Une couche de carte satellite (Mapbox Satellite) est ajoutée pour une référence visuelle plus détaillée.

- Une mini-carte est intégrée pour permettre à l'utilisateur de garder une vue d'ensemble pendant l'exploration.
- La position de la souris est affichée en bas à gauche de la carte.

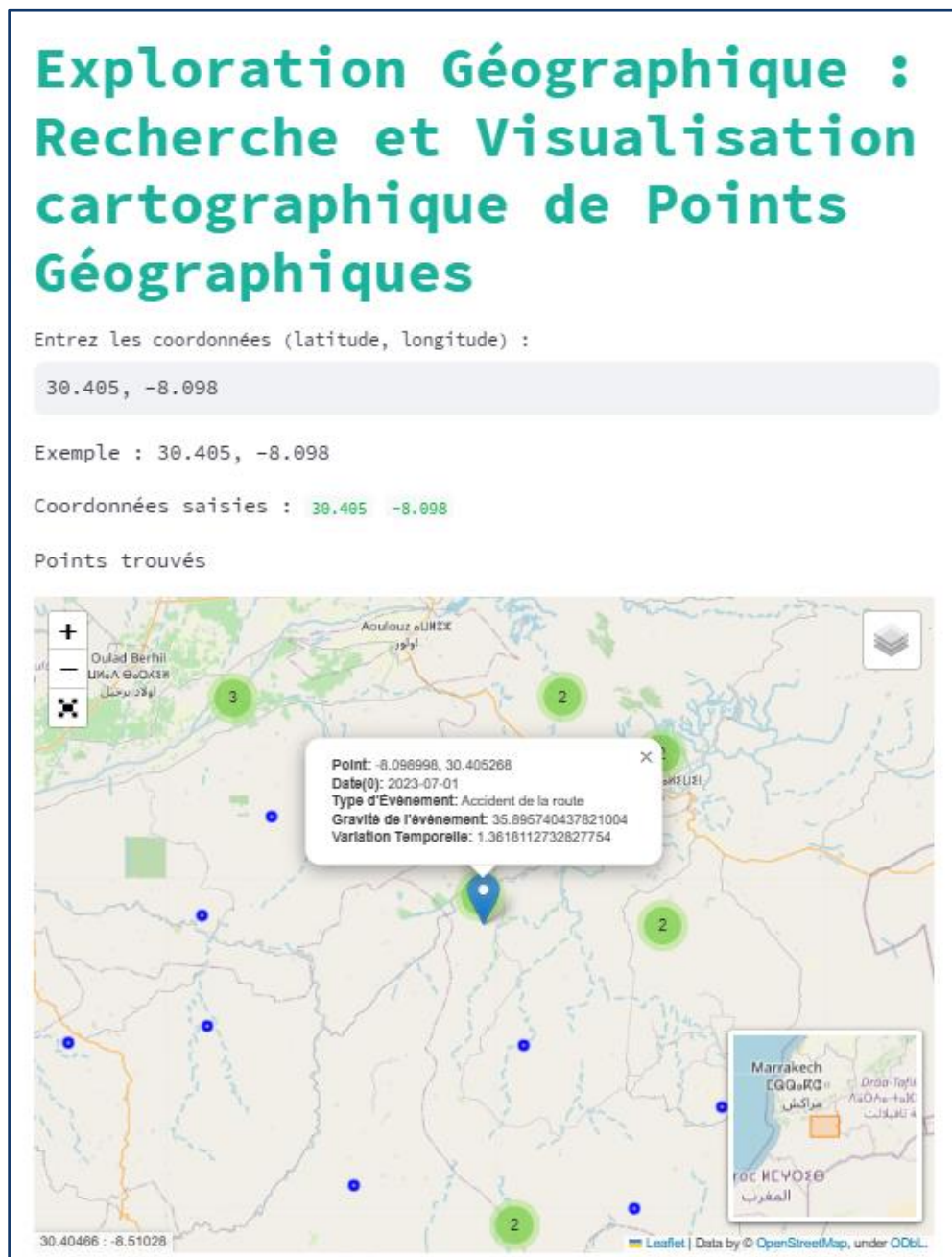
## 6. Contrôles Interactifs :

- Un contrôle de commutation de couche permet à l'utilisateur de choisir entre la carte standard et la carte satellite.
- Le contrôle de plein écran permet d'optimiser l'expérience visuelle.

## 7. Affichage avec Streamlit :

La carte interactive est affichée dans l'interface Streamlit à l'aide de **folium\_static()**

Résultat de l'application :



## g) Requêtes spatiales :

Dans cette partie nous avons fait un code qui crée une interface interactive permettant à l'utilisateur de filtrer et d'explorer des données géographiques stockées dans un GeoDataFrame en spécifiant la latitude min et max, et la longitude min et max. Il affiche une carte interactive avec des marqueurs regroupés, chacun fournissant des informations détaillées sur des événements géographiques spécifiques au sein de la zone filtrée.

### ➡ Description rapide du code :

#### 1. Importation des Bibliothèques :

#### 2. Chargement des données :

- Un GeoDataFrame (**gdf**) est créé en chargeant les données à partir d'un fichier GeoParquet.

#### 3. Filtrage des Données :

- Les données sont filtrées en fonction des bornes spatiales définies par l'utilisateur.
- Un GeoDataFrame filtré (**filtered\_gdf**) est créé.

#### 4. Vérifications sur les Données Filtrées :

- Si le GeoDataFrame filtré n'est pas vide, la carte interactive est créée.
- Sinon, des avertissements sont affichés, indiquant qu'aucune donnée n'est disponible pour les critères de filtre sélectionnés.

#### 5. Création de la Carte Interactive avec Folium :

- Une carte est créée avec Folium, centrée sur la zone sélectionnée.
- MarkerCluster de Folium est utilisé pour regrouper les marqueurs géographiques.
- Des marqueurs sont ajoutés à la carte, chacun associé à un popup contenant des informations sur le point géographique.

#### 6. Ajouts Supplémentaires à la Carte :

- Une mini-carte est ajoutée pour une vue d'ensemble.
- Une couche de carte satellite (Mapbox Satellite) est également ajoutée.
- Des contrôles interactifs comme la position de la souris, le plein écran, le contrôle des couches et la mesure sont incorporés.

#### 7. Affichage dans Streamlit :

- La carte interactive Folium est affichée dans l'interface Streamlit à l'aide de **folium\_static**.

- Résultat de l'application :

## Découvrez les données géographiques à l'aide de filtres interactifs et observez les marqueurs sur la carte.

Utilisez les contrôles ci-dessous pour ajuster les coordonnées spatiales et explorer les événements géographiques.

Latitude minimale

21.31

- +

Latitude maximale

21.31

- +

Longitude minimale

-16.99

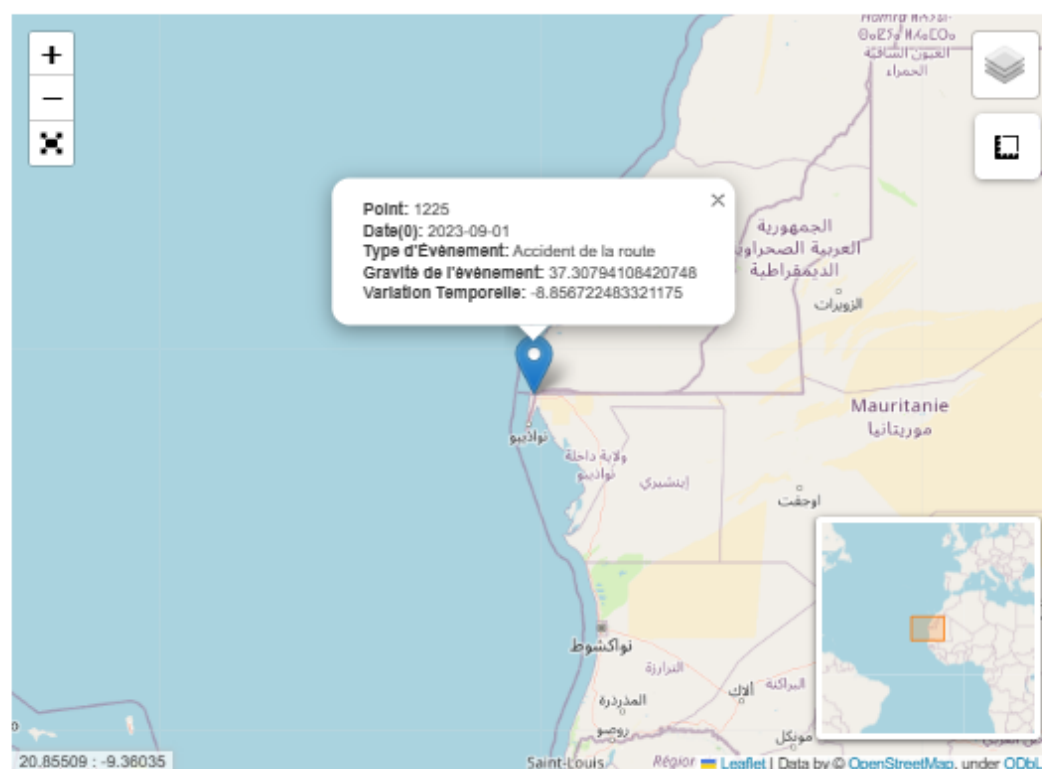
- +

Longitude maximale

-16.99

- +

Carte filtrée avec informations sur chaque point :



## h) Exploration de l'utilisation du format COG pour les cartes

### Raster :

Le format COG (Cloud Optimized Geotiff) est une évolution du format Geotiff pour les données Raster. Il présente plusieurs différences et avantages par rapport au format Geotiff traditionnel.

L'objectif de cette exploration est d'analyser l'utilisation du format COG dans le contexte des cartes Raster et de mettre en évidence son importance par rapport au GeoTIFF classique.

#### Tableau comparatif :

COG	GeoTIFF
<ul style="list-style-type: none"><li>- Fichier géoréférencé</li><li>- Réduction de la taille des fichiers (ce qui facilite leur stockage, leur transfert et leur traitement.)</li><li>- Compatibilité avec les plateformes et les outils modernes de visualisation des données géospatiales.</li><li>- Possibilité d'accéder aux données géospatiales à partir de navigateurs web et d'applications de visualisation.</li><li>- Facilité d'intégration avec les services cloud et les infrastructures de données géospatiales.</li><li>- Mise à jour progressive : permet une mise à jour progressive des données raster, ce qui signifie que les modifications peuvent être apportées aux fichiers COG sans avoir à recréer complètement les données</li></ul>	<ul style="list-style-type: none"><li>- Fichier géoréférencé</li><li>- Grande taille</li><li>- Utiliser pour le stockage des données raster</li><li>- Inclut des métadonnées détaillées sur l'image</li></ul>

#### Utilisation du format COG :

Le format COG est de plus en plus utilisé dans différents scénarios liés aux données géospatiales :

- **Analyse Géospatiale** : Le format COG permet une analyse plus efficace des données géospatiales en permettant un accès rapide aux tuiles d'image sans avoir besoin de télécharger l'ensemble du fichier.



- **Visualisation Interactive** : Le format COG permet une visualisation interactive des données géospatiales, en permettant le chargement rapide des tuiles d'image et la navigation fluide à travers les différentes résolutions.
- **Diffusion de Données Géospatiales** : Le format COG facilite la diffusion de données géospatiales en permettant un accès rapide et efficace aux images raster, ce qui est particulièrement utile pour les applications en ligne et les services de cartographie.

### Dans notre cas :

Après la conversion de nos cartes Raster depuis le format GeoTIFF vers le format Cloud-Optimized GeoTIFF (COG), nous avons observé une réduction significative de la taille des fichiers. Les fichiers individuels sont passés d'une moyenne de 9 à 10 Mo à seulement 0,9 à 1 Mo. Cette transformation a également eu un impact notable sur la taille totale du dossier contenant ces cartes, réduisant considérablement son encombrement de plus de 500 Mo à seulement 63 Mo. Cette optimisation a facilité le stockage des données sur notre GitHub, permettant un processus d'importation rapide et fluide.

	cartes interpolées
Type:	File folder
Location:	D:\3CI TOPO\Web Mapping\Projet\Partie1\code
Size:	587 MB (616 111 786 bytes)
Size on disk:	587 MB (616 443 904 bytes)
Contains:	126 Files, 0 Folders

Figure 5 : Cartes Raster format GeoTIFF



	cartes interpolées_cog
Type:	File folder
Location:	D:\3CI TOPO\Web Mapping\Projet\Partie1\code
Size:	63,6 MB (66 752 624 bytes)
Size on disk:	63,7 MB (66 879 488 bytes)
Contains:	64 Files, 0 Folders

Figure 6 : Cartes Raster format COG

Bien que la raison principale derrière l'adoption du format Cloud-Optimized GeoTIFF (COG) résidait dans sa capacité intrinsèque à être intégré au cloud. L'idée était de tirer parti de cette fonctionnalité pour faciliter l'accès distant à nos données géospatiales sans nécessité de stockage local. Cependant, le problème est que la plupart des plateformes cloud sont payantes, nous avons dû renoncer à cette option. Nous avons donc choisi d'utiliser le COG localement, stockant nos données géospatiales sur nos propres serveurs. Malgré le choix moins répandu du stockage local, le COG a représenté une alternative pragmatique pour nos besoins en gestion de données géospatiales. Par la suite, constatant les avantages de cette approche, nous avons opté pour l'utilisation des cartes au format COG dans nos scripts python.

En conclusion, il est recommandé d'adopter le format COG pour une meilleure efficacité et compatibilité des cartes Raster. Son utilisation permettra de faciliter l'accès aux données, d'améliorer les performances et de favoriser la collaboration entre les utilisateurs.



## i) Export du contenu du Dashboard :

Pour l'export du contenu du Dashboard, on a utilisé un code qui crée une interface permettant à l'utilisateur de télécharger le fichier PDF. Il affiche un bouton, et lorsque l'utilisateur clique sur ce bouton, un lien de téléchargement pour le fichier PDF préexistant est généré.

Streamlit ne propose pas de fonction native pour exporter le contenu des pages au format PDF. Pour obtenir le contenu des pages sous ce format, on peut : imprimer la page directement ou préconfigurer un fichier PDF contenant le contenu des pages pour être téléchargé par les utilisateurs. Comme on a fait dans notre code.

Nous avons utilisé le PDF de notre rapport de projet et non pas un PDF qui englobe tout le contenu de notre Dashboard.

### ➡ Description rapide du code :

#### 1. Importation des Bibliothèques :

#### 2. Fonction de Génération de Lien de Téléchargement :

- La fonction `create_download_link` prend en paramètre le chemin vers un fichier PDF et génère un lien de téléchargement.
- Elle ouvre le fichier PDF en mode lecture binaire, encode les données en base64, et crée un lien hypertexte avec l'encodage base64.
- Le label du lien est personnalisable, par défaut, il est défini comme "Télécharger le PDF".

#### 3. Chemin vers le Fichier PDF :

- Définition du chemin vers le fichier PDF.

#### 4. Configuration de l'Application Streamlit :

- Le titre de l'application est défini comme une chaîne vide pour des raisons de présentation.
- Un titre HTML est ajouté avec une mise en forme spécifique pour annoncer l'objectif de l'application.

#### 5. Bouton de Téléchargement :

- Un bouton Streamlit avec l'étiquette "Télécharger le PDF" est créé.

#### 6. Affichage du Lien de Téléchargement :

- Lorsque le bouton de téléchargement est cliqué, un lien hypertexte est généré à l'aide de la fonction `create_download_link`.

- Le lien est affiché dans l'application Streamlit.

Résultat de l'application :

# Exporter le contenu de notre dashboard.

Télécharger le PDF