

Rapport du projet chef d'œuvre  
Tableau de bord pour le suivi de l'épidémie Covid-19

—  
Certification Développeuse Data



Date	Révision	Rédigé par	Validé par
14 mai 2021	13 mai 2021	Lamia ADJIR	—

# Sommaire

0.0	Introduction . . . . .	1
<b>I</b>	<b>Analyse de la demande :</b>	<b>2</b>
0.1	Problématique . . . . .	2
0.2	État de l'art . . . . .	2
0.3	Description fonctionnelle de notre projet . . . . .	3
0.3.1	Le schéma fonctionnel : . . . . .	4
<b>II</b>	<b>Mise en œuvre du projet</b>	<b>5</b>
0.1	Gestion du projet . . . . .	5
0.2	Origine des données . . . . .	6
0.3	Conception Technique . . . . .	8
0.3.1	Description Statique du système . . . . .	8
0.3.2	Description dynamique du système . . . . .	9
0.4	Architecture technique détaillée . . . . .	10
0.4.1	Le choix Technologique . . . . .	10
0.4.1.1	Base de données relationnelles ou Nosql ? . . . . .	10
0.4.1.2	L'utilisation d'un ORM : . . . . .	11
0.4.1.3	Langages, bibliothèques et les outils utilisés . . . . .	12
0.4.2	Arborescence du COVID-APP . . . . .	12
0.5	Développement de la base de données . . . . .	13
0.5.1	Création de la base de données(Initializer) . . . . .	13
0.5.2	Alimentation de la base de données(Initializer) . . . . .	18
0.5.3	Mise à jour de la base de données(Updater) . . . . .	19
<b>III</b>	<b>Présentation de l'analyse et des résultats(FlaskPlotlyAPP)</b>	<b>20</b>
0.1	Représentation visuelle des données . . . . .	20

0.2	Présentation du Dashboard final . . . . .	23
-----	---	----

<b>IV</b>	<b>Conclusion générale et perspectives</b>	<b>28</b>
-----------	--	-----------

0.3	Conclusion générale et perspectives . . . . .	29
-----	---	----

0.4	Références . . . . .	30
-----	----------------------	----

## 0.0 Introduction

Jusqu'en 2002, les coronavirus n'étaient vu comme un problème que pour les personnes immunodéprimées et les nourrissons, susceptibles de développer des complications respiratoires de type pneumonie en cas d'infection. Pour les autres, au pire des cas, c'était paracétamol et mouchoirs !

Et puis il y a eu le Sars-CoV, un nouveau coronavirus apparu en Chine, qui a non seulement acquis le super pouvoir de se transmettre de l'animal à l'Homme puis d'Homme à Homme, mais aussi celui de déclencher une détresse respiratoire aigüe, voire le décès des personnes infectées. En 2012, rebelote avec Mers-CoV, apparu cette fois-ci en Arabie Saoudite.

Un troisième coronavirus agressif et transmissible à l'Homme a émergé en Chine mi-décembre 2019. Il s'agit d'un proche cousin du Sars-CoV, baptisé Sars-CoV2. La maladie qu'il entraîne est quant à elle nommée Covid-19. Il reste à ce jour de nombreuses inconnues quant à la biologie de ce virus. [1]

Depuis le 11 mars 2020, l'OMS qualifie la situation mondiale du COVID-19 de pandémie ; c'est-à-dire que l'épidémie est désormais mondiale. En un an, le virus a été contracté par plus de 100 millions de personnes et en a tué plus de 2,5 millions.

En France, où l'épidémie s'est étendue, l'exécutif a mis en place l'interdiction de déplacement, vulgarisée dans les médias par les expressions « confinement de la population », « confinement national » ou « confinement partiel », est une mesure sanitaire mise en place à trois reprises :

1. du 17 mars à 12 h au 11 mai 2020 (non inclus, soit 1 mois et 23 jours)
2. du 30 octobre au 15 décembre 2020 (non inclus, soit 1 mois et 14 jours)
3. du 3 avril au 3 mai 2021 (non inclus). [2]

Dans le cadre de la certification "Développeuse Data(Python) -Simplon Grand Ouest" , nous nous intéressons à l'évolution quotidienne de la pandémie en France en répondant aux questions suivantes :

- Comment exploiter efficacement les données recensées pour une meilleure visibilité de la propagation du virus ?
- Quelles sont les régions(départements) les plus / moins touchées ?
- Quel était l'effet des mesures sanitaires mises en place(confinement, couvre-feu..) ?
- Comment mesurer(comparer) les taux de létalité hospitalière (nombre de morts par rapport au nombre de patients hospitalisés) ?, qui probablement témoignent notamment de la performance des systèmes de soins.

## Première partie

### Analyse de la demande :

## 0.1 Problématique

La crise sanitaire due au Covid-19 a eu et aura des impacts importants sur notre société. Pour suivre l'évolution de cette situation et en mesurer les conséquences, la communauté scientifique et les institutions publiques peuvent s'appuyer entre autres sur les sources de données quantitatives, qu'il s'agisse de données d'enquêtes ou de données administratives.

En France, plusieurs organismes ont mis à disposition des données très détaillées et sensibles. Dans ce projet nous allons exploiter les données qui proviennent principalement de [data.gouv.fr](https://data.gouv.fr)<sup>1</sup> et de [OpenCOVID19-fr](https://opencovid19-fr.github.io)<sup>2</sup> dont l'objet est de consolider les données et proposer des outils de visualisation concernant l'épidémie de Covid-19 en France.

Dans cette perspective, comment pouvons-nous traiter ces données pour une représentation visuelle simple et intuitive ?

## 0.2 État de l'art

Pour répondre à une partie de la problématique citée dans la section précédente, nous trouvons sur internet plusieurs outils de visualisation. Dans cette partie nous avons choisi de vous présenter un tableau de bord dynamique proposé par Data-gouv-fr.

Ce tableau de bord présente les données relatives à l'épidémie de Covid-19 en France. Il a été mis en ligne le 28 mars 2020, afin de répondre aux besoins d'information des citoyens et des citoyennes en matière de transparence sur l'évolution de l'épidémie.

L'ensemble des données publiées sur ce tableau de bord sont consolidées par le Ministère des Solidarités et de la Santé et Santé publique France.

Cet outil, dont le code source est libre, est développé sous l'impulsion d'Etalab, au sein de la direction interministérielle du numérique.

Il propose une vision consolidée des données officielles disponibles, Plusieurs sources de données viennent l'alimenter à savoir :

**Données hospitalières** : Données hospitalières relatives à l'épidémie de COVID-19

**Données relatives aux tests**

**Données relatives aux vaccins**

**Données de la direction générale de la santé**

---

1. [data.gouv.fr](https://data.gouv.fr) est une plateforme de diffusion de données publiques de l'État français. [data.gouv.fr](https://data.gouv.fr) est développé par Etalab, une mission placée sous l'autorité du Premier ministre

2. Organisation informelle issue de la société civile, accessible sur <https://github.com/opencovid19-fr>

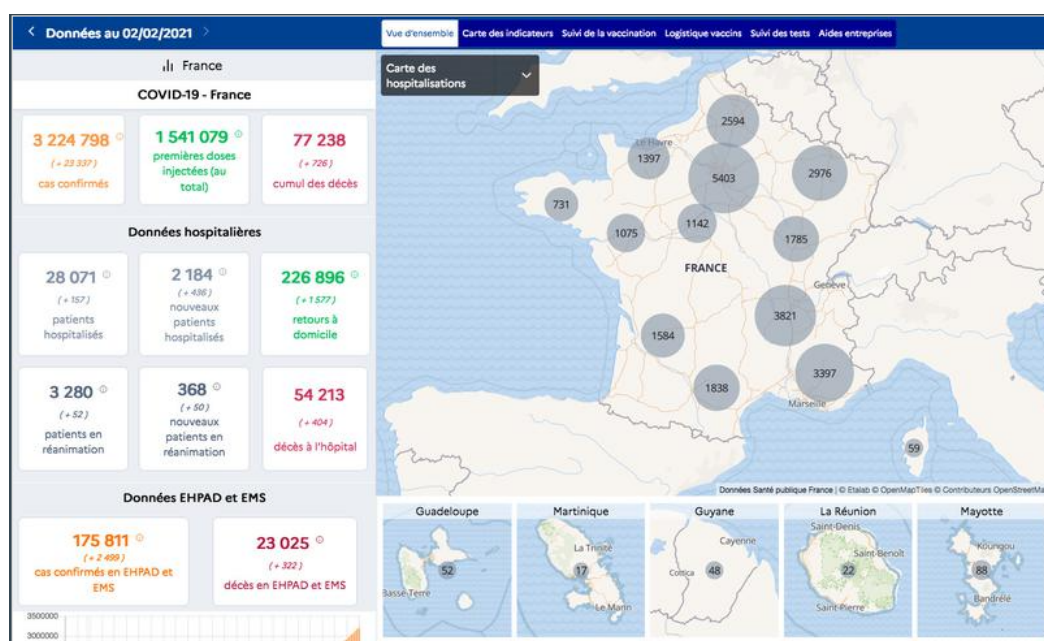
Tous les jours, la direction générale de la santé communique, à l'échelle nationale :

Le nombre de décès cumulés en EHPAD et EMS ;

Le nombre de cas confirmés en EHPAD et EMS.

**Le tableau de bord est disponible à l'adresse suivante :** <https://www.gouvernement.fr/info-coronavirus/carte-et-donnees>

**Pour plus de détails voir** <https://www.data.gouv.fr/fr/reuses/tableau-de-bord-de-suivi-de-lepidemie-de-coronavirus-en-france/>



Autant dire que cet outil répond à une partie de notre problématique à savoir la visualisation dynamique des données, cependant il ne couvre pas la partie analyse et statistique notamment l'évaluation de l'effet des mesures sanitaires sur la propagation du virus.

### 0.3 Description fonctionnelle de notre projet

Pour atteindre nos objectifs, nous avons été amenée à développer une application web sous le nom de "COVID-APP", cette application va permettre à l'utilisateur d'une part de consulter la dernière mise à jour des données, d'une autre part, elle va lui donner la possibilité d'accéder aux statistiques et analyses réalisées.

Dans ce rapport, nous allons vous présenter les différentes étapes qui ont conduit à la réalisation de ce projet.

### 0.3.1 Le schéma fonctionnel :

Afin de décrire les fonctionnalités de ce système(Covid-App), Nous entamons d'abord par présenter ses acteurs :

- Visiteur : rôle joué par tout visiteur étranger au système.
- Utilisateur : rôle joué par toute personne s'étant inscrite dans le système.
- Administrateur : responsable de la gestion des utilisateurs(Ajout, suppression, modification des utilisateurs.)

Ensuite, nous allons vous présenter le schéma fonctionnel sous format de cas d'utilisation <sup>3</sup>

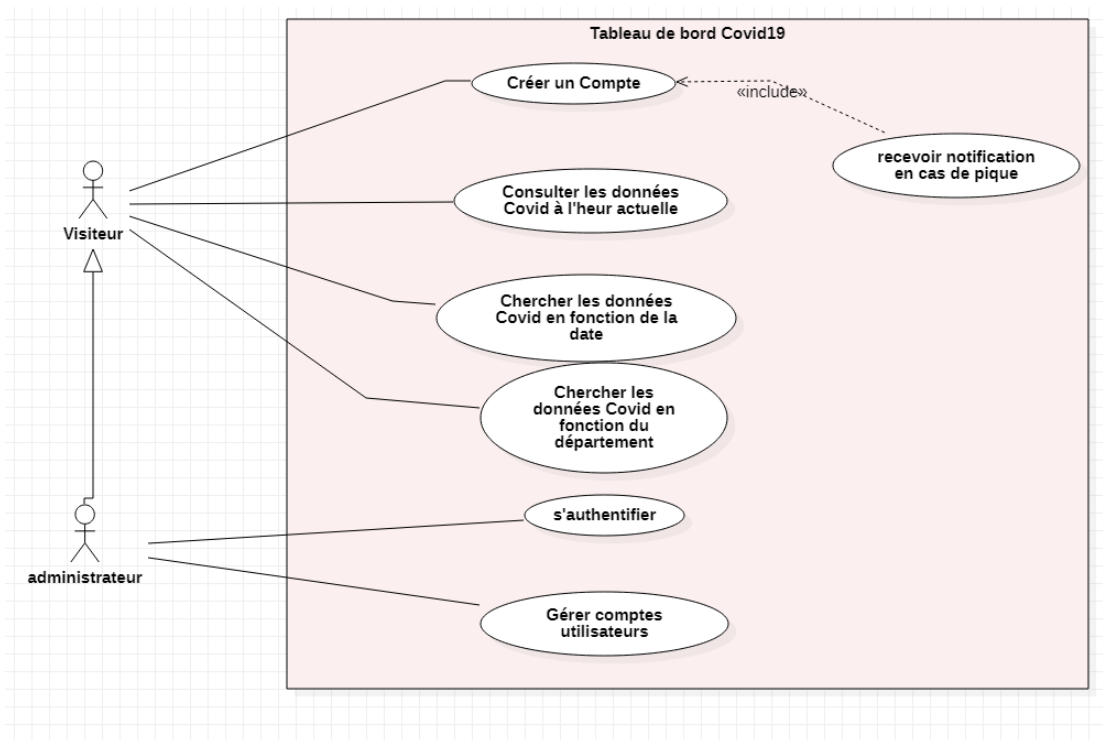


FIGURE 1 – Diagramme Cas d'utilisation

- Un visiteur ou un administrateur peut créer un compte sur le système.
- Un visiteur peut consulter les résultats Covid-19 à l'heure actuelle.
- Un visiteur a la possibilité d'effectuer une recherche en choisissant la date souhaitée.
- Un utilisateur peut consulter les statistiques et les analyses les plus détaillées.
- un administrateur a le droit de créer des comptes utilisateur, ou les supprimer.

3. cas d'utilisation : Il s'agit de la solution UML pour représenter le modèle conceptuel. Les cas d'utilisation (ou use cases en Anglais) permettent de structurer les besoins des utilisateurs et les objectifs correspondants d'un système.



## Deuxième partie

### Mise en œuvre du projet

## 0.1 Gestion du projet

Notre projet étant un projet individuel dont le sujet est libre de choix. La première étape a donc été de bien réfléchir pour déterminer le contexte et le sujet du projet.

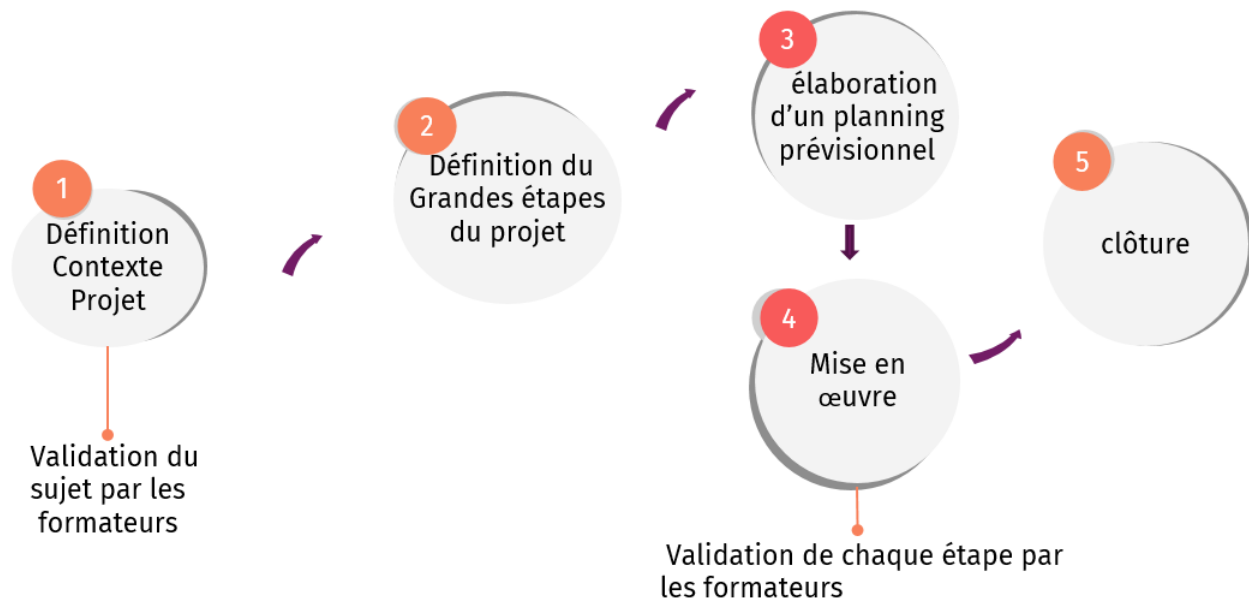


FIGURE 2 – Déroulement logique du projet

Ce schéma permet de décrire l'organisation générale d'un projet en définissant la logique de son déroulement. L'objectif étant de faire une première planification prévisionnelle du projet en le découpant en grandes étapes et de faire valider chaque étape par les formateurs.

Pour l'organisation et le suivi du projet, après avoir raffiné les grandes étapes de projet en les découpant en tâches plus petites et réalisables, nous avons choisi d'encadrer notre projet par la méthode Kanban,

Pour réaliser notre tableau Kanban nous nous sommes appuyée sur l'outil « Birix24 ». birix24 est un outil de gestion de projet en ligne, la façon la plus simple de configurer notre tableau Kanban personnel était de créer trois colonnes : À faire, En cours, Terminé . Comme le montre la figure juste après.

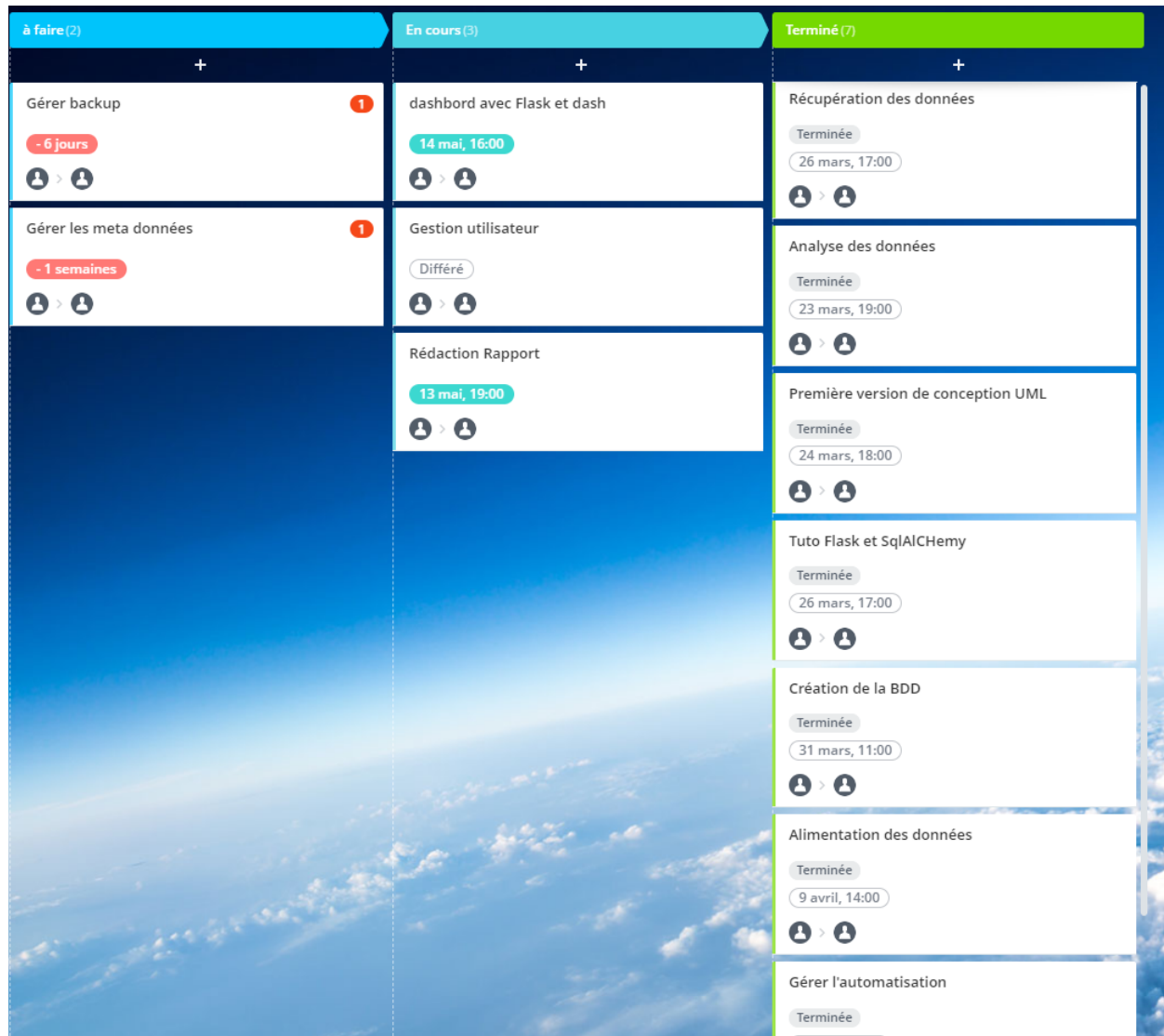


FIGURE 3 – Tableau Kanban

## 0.2 Origine des données

Pour la réalisation de ce projet, nous avons exploité trois types de données à savoir :

### 1. *Le jeu données Format CSV :*

Publié sur data.gouv.fr le 15 août 2019 et mis à jour le 15 août 2019.

Ce fichier contient les informations suivantes :

- num\_dep : correspond au numéro du département.
- dep\_name : correspond au nom du département.
- region\_name : correspond à de la région de ce département.

L'objectif de ce jeu de données est de permettre l'obtention rapide de la région correspondant à un département français.

## 2. *Le jeu données Format JSON :*

Ce jeu de données provient de l'initiative OpenCOVID19-fr<sup>4</sup>, ces données sont proposées selon plusieurs granularité : monde, pays(France), région et département.

### **Parmi les différentes sources utilisées :**

- Santé publique France
- Préfectures
- Agences Régionales de Santé
- Ministère des Solidarités et de la Santé

### **Parmi les données nationales :**

- casConfirmes : total cumulé du nombre de cas confirmés
- deces : total cumulé du nombre de décès
- decesEhpad : total cumulé du nombre de décès en EHPAD (si indiqué)
- hospitalises : nombre de personnes hospitalisées le jour du bulletin
- reanimation : nombre de personnes en réanimation le jour du bulletin
- gueris : total cumulé du nombre de personnes guéries (sorties de l'hôpital)
- NouvellesReanimation et NouvelleHospitalisation.

Voici un exemple de bloc du fichier Chiffres-clés.json :

```
{  "code": "DEP-66",
  "nom": "Pyrénées-Orientales",
  "date": "2020-03-21",
  "hospitalises": 25,
  "reanimation": 15,
  "nouvellesHospitalisations": 1,
  "nouvellesReanimations": 1,
  "deces": 5,
  "gueris": 3,
  "source": {
    "nom": "Santé publique France Data"
  },
  "sourceType": "sante-publique-france-data"}
```

4. Données nationales concernant l'épidémie de COVID19 accessible sur :  
<https://raw.githubusercontent.com/opencovid19-fr/data/master/dist/chiffres-cles.json>

### 3. Le jeu données *CoronavirusAPI-France* :

API permettant de récupérer et exploiter les données sur le Coronavirus en France actualisées chaque jours à 20H pour chaque département et globalement pour la France entière via de simple requêtes GET qui enverront une réponse en JSON

## 0.3 Conception Technique

### 0.3.1 Description Statique du système

Pour décrire clairement la structure du système COVID-APP nous avons opté pour la réalisation d'un diagramme de Classes(UML)

Ce diagramme permet le modélisation des classes du système, ses attributs, ses opérations et les relations entre ses objets.

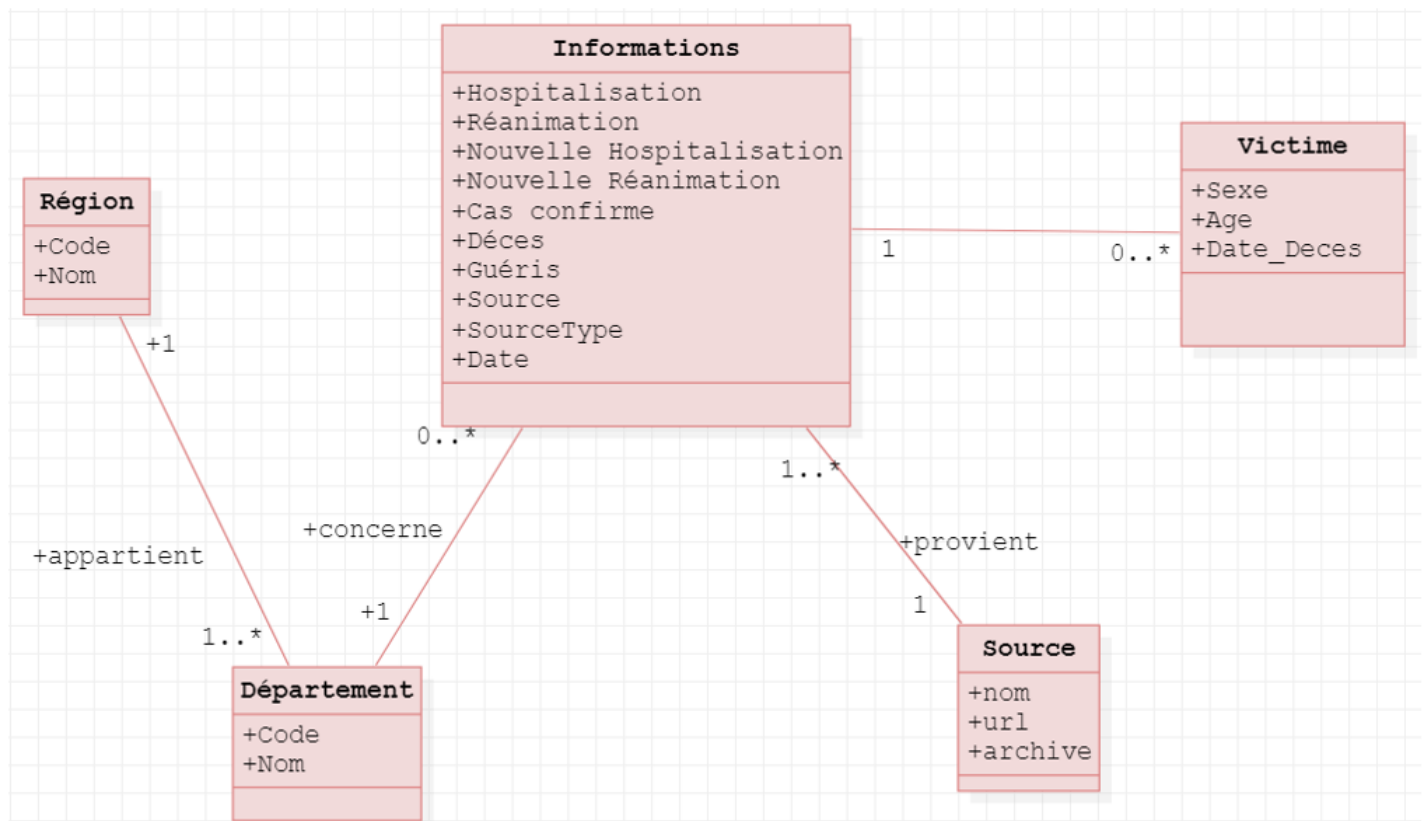


FIGURE 4 – Diagramme de Classes

Covid-APP contient 5 Classes à savoir :

- Département : contient Code et nom du département.
- Région : Contient le nom et code de la région.
- Information : regroupe les données relatives au Covid-19( hospitalisation, réanimation, décès..).
- Source : représente la source de l'information.
- Victime : décrit le sexe, l'âge et la date de décès.

Les relations entre ses classes sont les suivantes :

- Un département appartient à une région, une région contient plusieurs départements.(OneToMany)
- Une information concerne un département, par transitivité une information concerne aussi une région. (**OneToMany**)
- Un objet Information provient d'une seule source, une source peut représenter plusieurs informations. (**OneToMany**)

### 0.3.2 Description dynamique du système

Pour décrire les différents interactions entre les différentes parties du Covid-APP nous allons vous présenter le diagramme de séquence<sup>5</sup> :

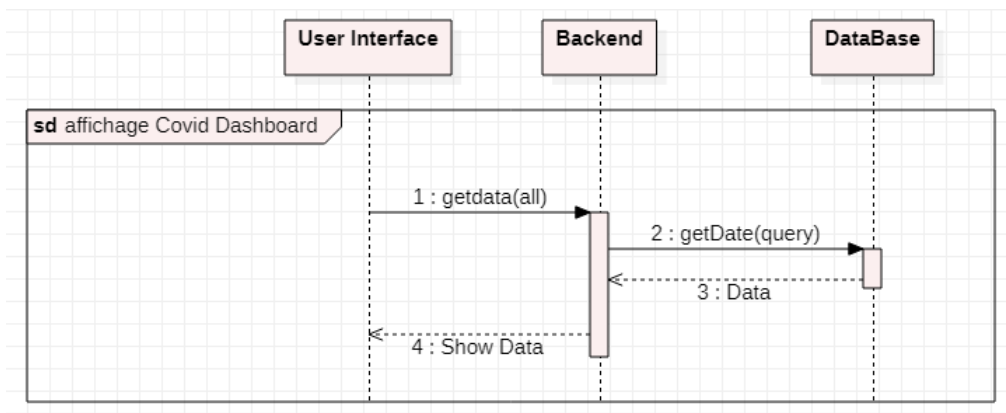


FIGURE 5 – Diagramme de séquence : Affichage Covid Dashboard

La figure ci-dessus montre les interactions entre l'interface utilisateur(cela peut-être un navigateur web), le backend et la base de données.

- Dans un premier temps, l'utilisateur interagit avec le navigateur pour consulter les l'interfaces du Covid-APP.
- Le système à son tour va interroger la BDD pour récupérer cette information et renvoi la réponse à l'utilisateur.

5. Diagramme de séquence permet de décrire les scénarios de chaque cas d'utilisation en mettant l'accent sur la chronologie des opérations en interaction avec les objets.

La figure juste après décrit le scénario de mise à jour des données ainsi que l'interaction entre Batchfile, ApiCoronaVirusApi et BDD.

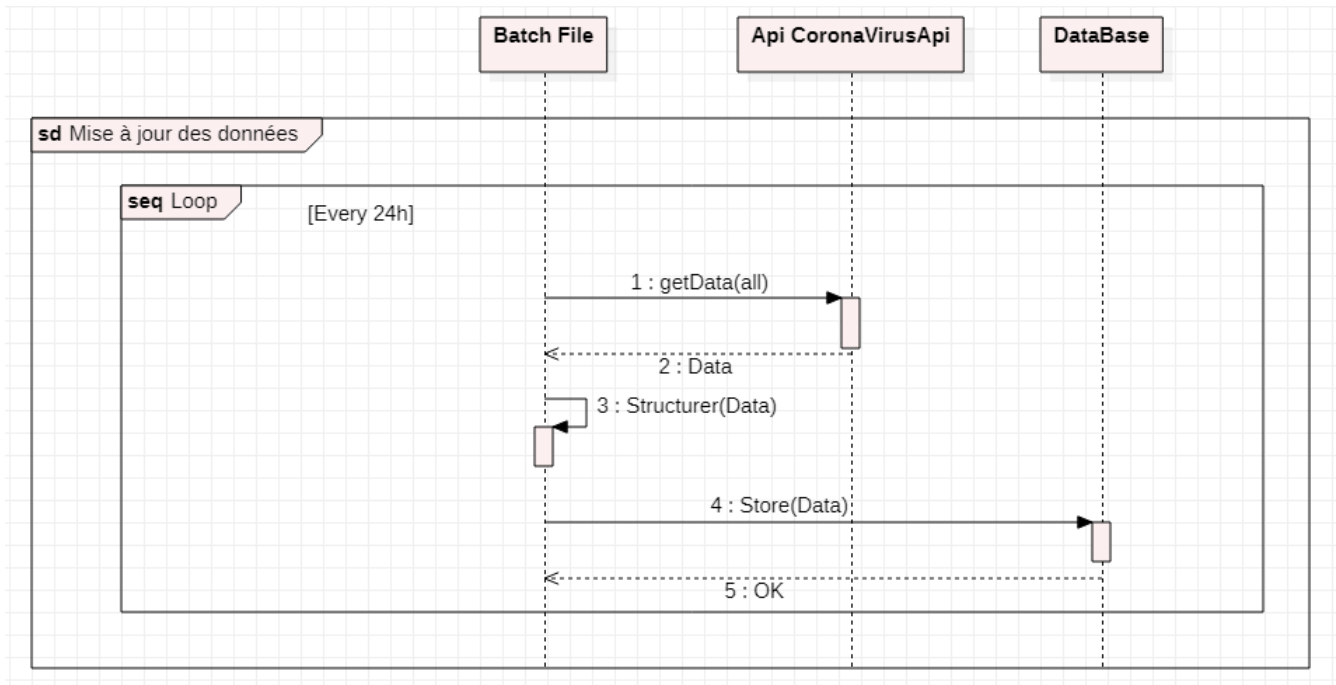


FIGURE 6 – Diagramme de séquence : Mise à jour des données

Chaque jour, une tâche Windows, exécute un script python écrit sur un fichier batch(automate.bat), ce script va interroger CoronavirusAPI-France pour récupérer les dernières mises à jour et les sauvegarder dans la BDD.

## 0.4 Architecture technique détaillée

### 0.4.1 Le choix Technologique

#### 0.4.1.1 Base de données relationnelles ou Nosql ?

Dans ce projet nous avons choisi de sauvegarder nos données sur une base de données relationnelle "postgres".

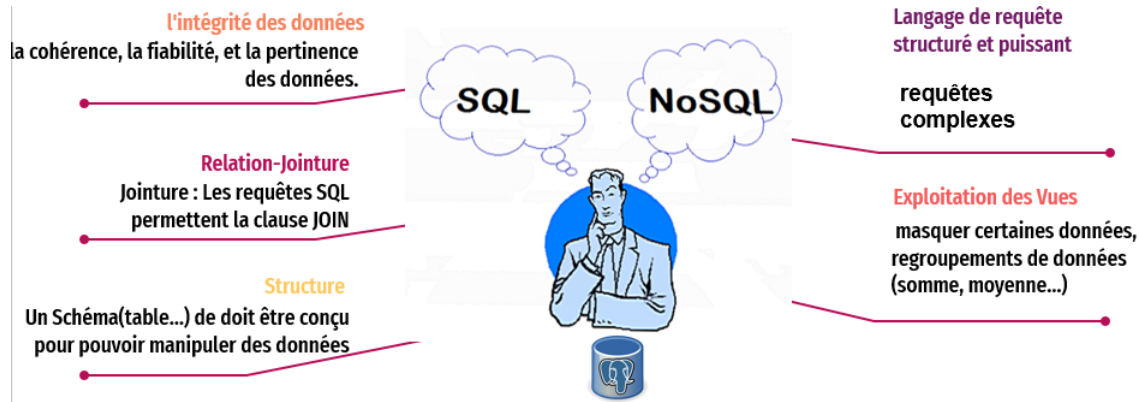


FIGURE 7 – BDD relationnelle ou Nosql ?

Les raisons principales de ce choix reviennent, premièrement à la possibilité de classer les données en créant des relations de jointures entre les tables tout en maintenant la cohérence et la fiabilité des données,

Les bases de données SQL permettent aussi l'application des requêtes complexes et l'exploitation du concept des vues.<sup>6</sup>

#### 0.4.1.2 L'utilisation d'un ORM :

Pour la réalisation de ce projet, nous avons utilisé l'ORM SqlAlchemy.<sup>7</sup>

L'ORM SqlAlchemy nous a permis de spécifier explicitement dans le code les relations entre les tables( OneToMany, ManyToMany... ) , ce qui facilite la gestion des clés étrangères et la syntaxe des requêtes. Pour plus de détails voir la figure 1.5.

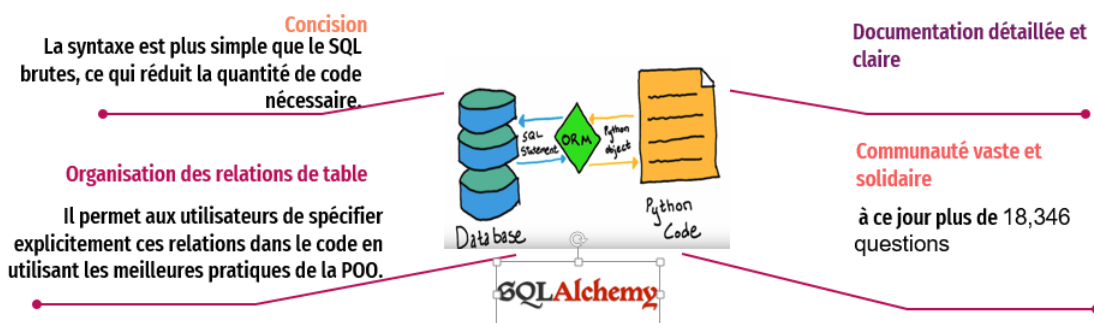


FIGURE 8 – justification choix technique

6. Une vue dans une base de données est une synthèse d'une requête d'interrogation de la base. On peut la voir comme une table virtuelle, définie par une requête.

7. SQLAlchemy est un toolkit open source SQL et un mapping objet-relationnel (ORM) écrit en Python. SQLAlchemy a été publié en février 2006 et est rapidement devenu l'un des ORM les plus utilisés par la communauté Python.



### 0.4.1.3 Langages, bibliothèques et les outils utilisés

Voici un tableau reprenant l'ensemble des langages informatiques utilisés, les différentes bibliothèques et les outils pour la mise en œuvre du projet.

Langages	Bibliothèques	Framework	Moteur de Template	IDE	Logiciel
Python	pandas	Flask	Jinja	Pycharm	DBeaver
Html	ORM sqlalchemy	PlotlybyDash(Écrit au-dessus de Flask)		Texmaker	
Css	Requests	Bulma(Css)			
	Flask_login				
	werkzeug.security				
	MiKTeX				

FIGURE 9 – Choix technologique

**Requests** : Requests est une bibliothèque HTTP Python, publiée sous la licence Apache 2.0. Le but du projet est de rendre les requêtes HTTP plus simples et plus conviviales. (Wikipédia)

**Flask\_login** : Flask-Login fournit la gestion de session utilisateur pour Flask. Il gère les tâches courantes de connexion, de déconnexion et de mémorisation des sessions de vos utilisateurs sur de longues périodes.[3]

**Werkzeug.security** : permet de Hasher un mot de passe

**Dash** : Dash est un framework Python productif pour la création d'applications d'analyse Web.Écrit au-dessus de Flask, Plotly.js et React.js, Dash est idéal pour créer des applications de visualisation de données avec des interfaces utilisateur hautement personnalisées en Python pur.[4]

**TexMaker** : Texmaker est un logiciel libre destiné à l'édition de documents LaTeX.(Wikipédia)

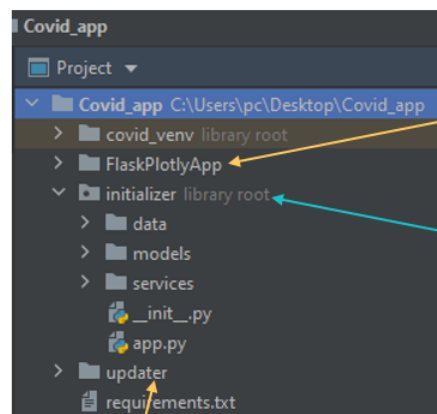
## 0.4.2 Arborescence du COVID-APP

Notre application est organisée sur 3 composants à savoir :

**Initializer** : permet la création de la base de données ainsi que son alimentation.

**Updater** : ce module assure la mise à jour de la BDD.

**FlaskPlotlyDash** : gère la partie Web ainsi que la Data Visualisation(Dashboard).



3

**FlaskPlotlyApp**

Dashboard et Data Vis

1

**Initializer**

- Création des tables
- Création des vues
- Alimenter la BDD

2

**Updater**

- Récupération des données de L'api France-Coronavirus
- Mise à jour de la BDD

FIGURE 10 – Structure de COVID-APP

## 0.5 Développement de la base de données

### 0.5.1 Création de la base de données(Initializer)

Dans "Inializer", Le fichier models.py contient la définition des classes région et département, information et source.

La classe victime n'a pas été implémentée sous la contrainte de manquement de données.

On exécutant ce fichier, l'ORM Ssqlalchemy va mapper les classes en tables, les attribues en colonnes.

```

class Region(Base):
    # Table name.
    __tablename__ = "region"
    # We always need an id
    code = Column(String, primary_key=True)
    name = Column(String(100), unique=True)
    departments = relationship("Department")

    def __init__(self, code, name):
        self.code = code
        self.name = name

    def __repr__(self):
        return "Region (code='%s', name='%s')" % (self.code, self.name)

class Department(Base):
    # Table name.
    __tablename__ = "departement"
    code = Column(String, primary_key=True)
    name = Column(String(100), unique=True)
    region = Column(String, ForeignKey('region.name'))
    information = relationship("Information")

    def __init__(self, code, name, region):
        self.code = code
        self.name = name
        self.region = region

    def __repr__(self):
        return "(code='%s', name='%s', region='%s')" % (self.code, self.name, self.region)

```

FIGURE 11 – Exemple du fichier Models.py

Voici les tableaux reprenant la description des tables et vues (models.py et CreatViews.py) :

Tables	Description	Clé Primaire	Clé étrangère (Table de Référence)
Source	La source de l'information	ID	—
Region	l'ensemble de régions françaises	code	—
Departement	l'ensemble de départements français	code	region(Region)
Role	décrit les Rôles de la l'application à savoir :Utilisateur, administrateur	ID	—
User	sauvegarde l'email, le nom et le passe de l'utilisateur	ID	—
roles_user	table de correspondance(Many-To-Many)	(ID_user,ID_role)	(ID_user (User), ID_role(Role))

TABLE 1 – Description des tables

Vues	Avantage
<b>CleanDataByDepartment :</b> Informations Covid19 groupées par département	<ul style="list-style-type: none"> <li>- Masquer des champs dont nous n'aurons pas besoin.</li> <li>- Traiter le cas où nous avons pour le même département, deux objets informations(même date)de deux sources différentes ;</li> </ul> Dans ce cas nous combinons ces deux objets en un seul en prenant le maximum de leurs valeurs, ce qui nous a permis aussi d'éliminer les doublons
<b>CleanDataByRegion :</b> Informations Covid19 par régions	Dans cette vue nous appliquons la somme des données sur l'ensemble des départements de chaque région.

TABLE 2 – Description des vues

```

metadata = MetaData(engine)
view = Table('CleanDataByDepartement', metadata)
definition = text('select i."date" as date,'
                  'i.nom as departement,'
                  'max(i.hospitalises) as hospitalises,'
                  'max(i.reanimation) as reanimation,'
                  'max(i."nouvellesHospitalisations") as "nouvellesHospitalisations",'
                  'max(i."nouvellesReanimations") as "nouvellesReanimations",'
                  'max(i.deces) as deces,'
                  'max(i."decesEhpad") as "decesEhpad",'
                  'max(i.gueris) as gueris,'
                  'max(i."casConfirmes") as "casConfirmes"'
                  'from information i '
                  'group by date, departement '
                  'order by date desc')

# Info by department
create_view = CreateView(view, definition, or_replace=True)

# Info by region
CleanDataByRegion = Table('CleanDataByRegion', metadata)
definition = text('select dp.date,d.region,r.code, '
                  'sum(dp.hospitalises) as hospitalises, '
                  'sum(dp.reanimation) as reanimation, '
                  'sum(dp."nouvellesHospitalisations") as "nouvellesHospitalisations", '
                  'sum(dp."nouvellesReanimations") as "nouvellesReanimations", '
                  'sum(dp.deces) as deces,'
                  'sum(dp."decesEhpad") as "decesEhpad",'
                  'sum(dp.gueris) as gueris,'
                  'sum(dp."casConfirmes") as "casConfirmes"'
                  'from public."CleanDataByDepartement" dp '
                  'join departement d '
                  'on dp.departement = d."name"'
                  'join region r on d.region = r."name"'
                  'group by date, d.region, r.code '
                  'order by date desc')
create_view = CreateView(CleanDataByRegion, definition, or_replace=True)
engine.execute(create_view)

```

FIGURE 12 – Exemple du fichier CreatViews.py

Voici le schéma de notre base de données finale :

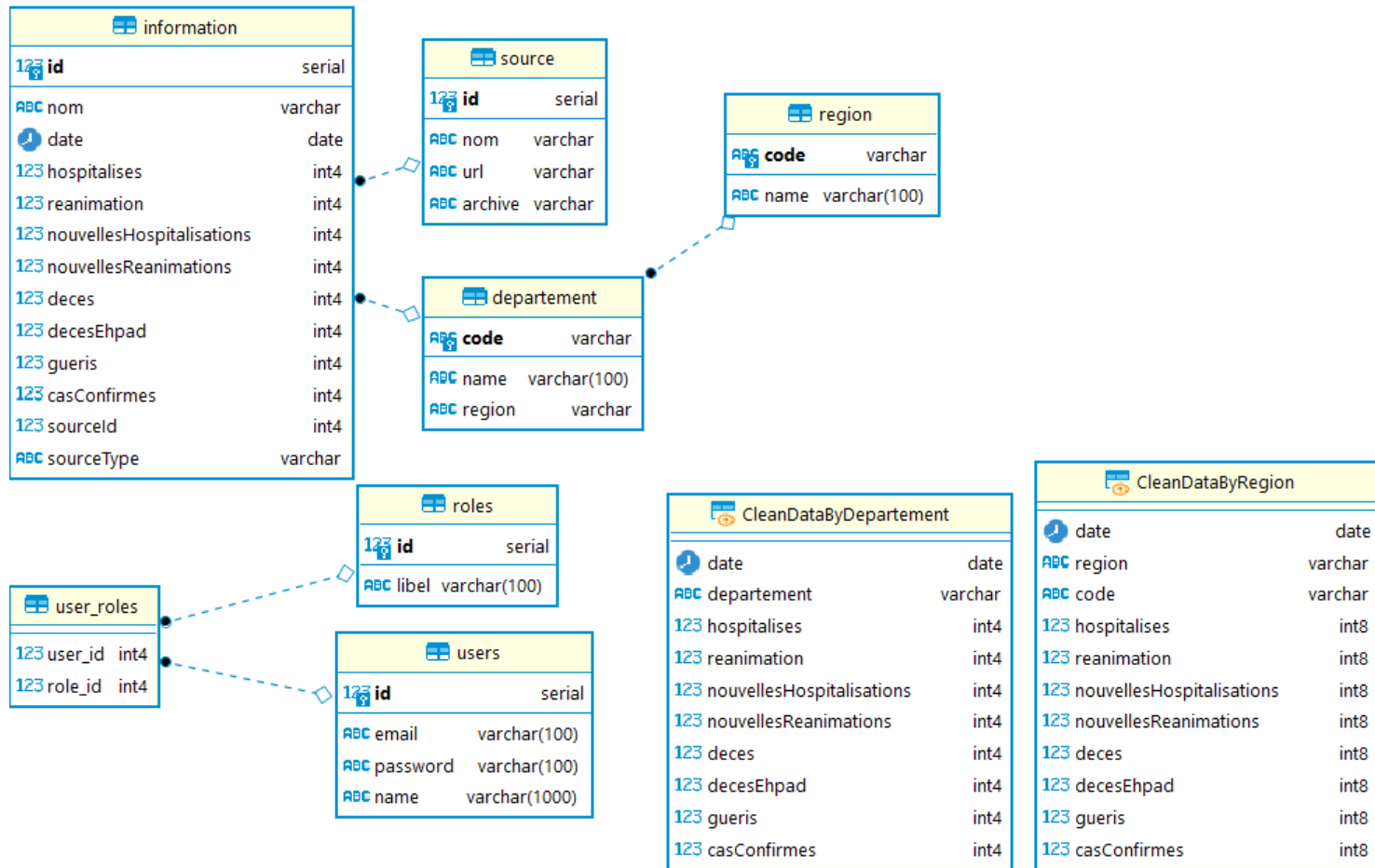


FIGURE 13 – ER Diagramme

### 0.5.2 Alimentation de la base de données(Initializer)

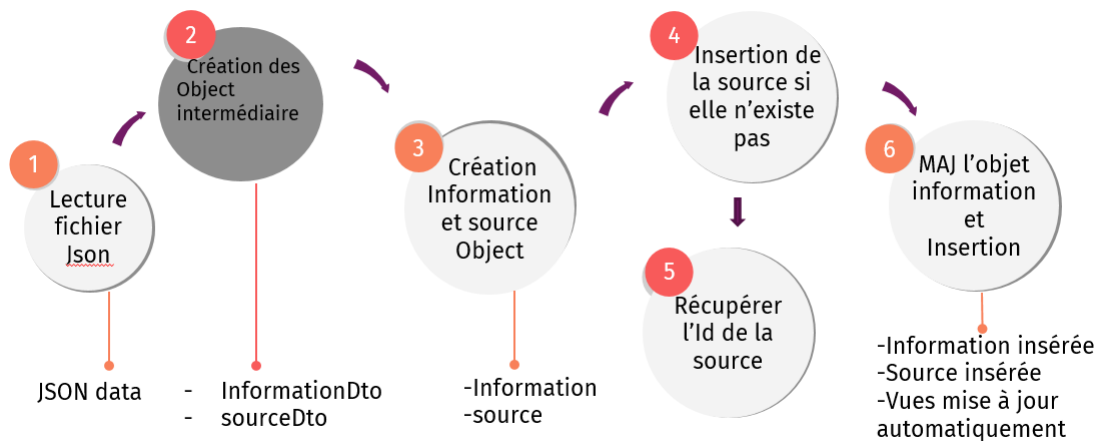


FIGURE 14 – Déroulement de l'alimentation de la BDD

L'alimentation de notre base de données passe par 5 étapes à savoir : - Lecture du fichier Chiffres-cle.json.

- Pour chaque informations récupérée, Nous en gardons que celles par département , nous avons choisi de ne pas récupérer les informations par région car elles sont calculables (Voir vues CleanDataByRegion).

- Création d'objet Information et Source.

- Insertion de la source si elle n'existe pas dans la BDD.

- Dans tous les cas nous récupérerons l'ID de la source.

- nous mettons à jour l'objet information en ajoutant le bon ID de la source comme clé étrangère.

- Mapper l'objet information en enregistrement dans la BDD.

### 0.5.3 Mise à jour de la base de données(Updater)

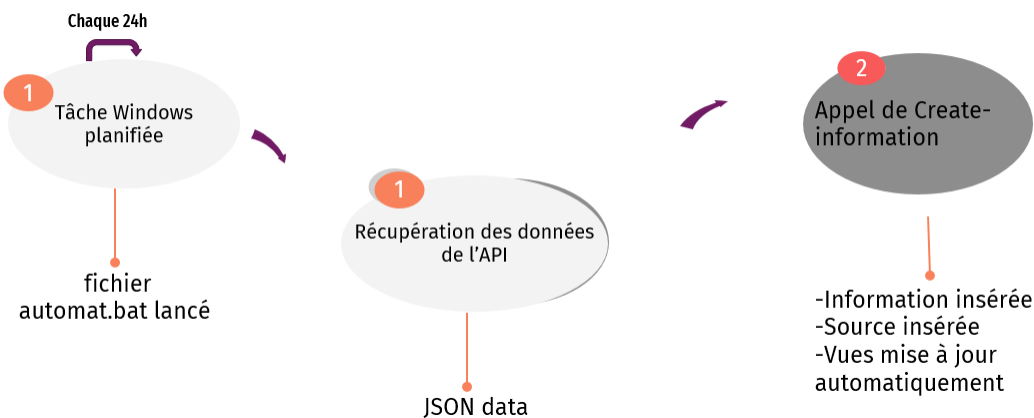


FIGURE 15 – Déroulement de la mise à jour de la BDD

La mise à jour de notre base de données passe par 3 étapes à savoir :

- Chaque 24h, une tâche Windows à été planifiée pour exécuter un script python,
- Ce script utilise la librairie "Requests" pour faire appel à l'api CoronavirusAPI-France
- Une fois que les données sont récupérées, nous appelons le script qui permet l'insertion dans la BDD.

Voici un exemple d'exécution de ce script :

```

C:\WINDOWS\SYSTEM32\cmd.exe
2021-05-06 09:45:24,881 INFO sqlalchemy.engine.base.Engine {'nom': 'OpenCOVID19-fr', 'param': 1}
2021-05-06 09:45:24,885 INFO sqlalchemy.engine.base.Engine INSERT INTO information (nom, date, hospitalises, reanimation, "nouvellesHospitalisations", "nouvellesReanimations", decs, "decsEhpad", gueris, "casConfirmes", "sourceId", "sourceType") VALUES (%(nom)s, %(date)s, %(hospitalises)s, %(reanimation)s, %(nouvellesHospitalisations)s, %(nouvellesReanimations)s, %(decs)s, %(decsEhpad)s, %(gueris)s, %(casConfirmes)s, %(sourceId)s, %(sourceType)s) RETURNING information.id
2021-05-06 09:45:24,885 INFO sqlalchemy.engine.base.Engine {'nom': 'Corse', 'date': '2021-05-05', 'hospitalises': 55, 'reanimation': 9, 'nouvellesHospitalisations': 3, 'nouvellesReanimations': 0, 'decs': 198, 'decsEhpad': None, 'gueris': 900, 'casConfirmes': None, 'sourceId': 9, 'sourceType': 'opencovid19-fr'}
2021-05-06 09:45:24,887 INFO sqlalchemy.engine.base.Engine ROLLBACK
error during insert
[{'code': 'DEP-01', 'nom': 'Ain', 'date': '2021-05-05', 'hospitalises': 180, 'reanimation': 18, 'nouvellesHospitalisations': 23, 'nouvellesReanimations': 0, 'decs': 594, 'gueris': 2678, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-02', 'nom': 'Aisne', 'date': '2021-05-05', 'hospitalises': 274, 'reanimation': 40, 'nouvellesHospitalisations': 15, 'nouvellesReanimations': 2, 'decs': 1043, 'gueris': 3837, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-03', 'nom': 'Allier', 'date': '2021-05-05', 'hospitalises': 93, 'reanimation': 19, 'nouvellesHospitalisations': 8, 'nouvellesReanimations': 1, 'decs': 558, 'gueris': 1966, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-04', 'nom': 'Alpes-de-Haute-Provence', 'date': '2021-05-05', 'hospitalises': 121, 'reanimation': 10, 'nouvellesHospitalisations': 5, 'nouvellesReanimations': 0, 'decs': 238, 'gueris': 1054, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-05', 'nom': 'Hautes-Alpes', 'date': '2021-05-05', 'hospitalises': 86, 'reanimation': 9, 'nouvellesHospitalisations': 2, 'nouvellesReanimations': 0, 'decs': 250, 'gueris': 1140, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-06', 'nom': 'Alpes-Maritimes', 'date': '2021-05-05', 'hospitalises': 435, 'reanimation': 66, 'nouvellesHospitalisations': 10, 'nouvellesReanimations': 3, 'decs': 1481, 'gueris': 6147, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-07', 'nom': 'Ardèche', 'date': '2021-05-05', 'hospitalises': 116, 'reanimation': 11, 'nouvellesHospitalisations': 3, 'nouvellesReanimations': 0, 'decs': 484, 'gueris': 1885, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-08', 'nom': 'Ardennes', 'date': '2021-05-05', 'hospitalises': 91, 'reanimation': 14, 'nouvellesHospitalisations': 0, 'nouvellesReanimations': 0, 'decs': 351, 'gueris': 1069, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}, {'code': 'DEP-09', 'nom': 'Ariège', 'date': '2021-05-05', 'hospitalises': 45, 'reanimation': 13, 'nouvellesHospitalisations': 0, 'decs': 198, 'gueris': 900, 'source': {'nom': 'Santé publique France Data'}, 'sourceType': 'sante-publique-france-data'}]
  
```

FIGURE 16 – Résultat d'exécution de la tâche planifiée



## Troisième partie

Présentation de l'analyse et des  
résultats(FlaskPlotlyAPP)

## 0.1 Représentation visuelle des données

[Home](#) [Profile](#) [Logout](#)

### CORONAVIRUS STATISTIQUES FRANCE – EVOLUTION DU NOMBRE DE CONTAMINATIONS JOUR PAR JOUR Date : 2021-05-05

1 2 « » Date Search

Region	Hospitalises	Reanimation	Nouvelles Hospitalisations	Nouvelles Reanimations	Décès	Guéris
Auvergne-Rhône-Alpes	3313	609	193	31	11120	47170
Bourgogne-Franche-Comté	1199	190	53	10	4587	18057
Bretagne	780	97	48	4	1535	7041
Centre-Val de Loire	1005	200	52	10	2558	10998
Corse	55	9	3	0	198	900
Grand Est	2335	467	107	28	9753	35251
Guadeloupe	159	30	30	4	252	1173
Guyane	75	17	11	2	99	2388
Hauts-de-France	3246	627	129	33	8534	33824

displaying 1 - 9 records in total 18

FIGURE 17 – Page d'accueil de COVID-APP

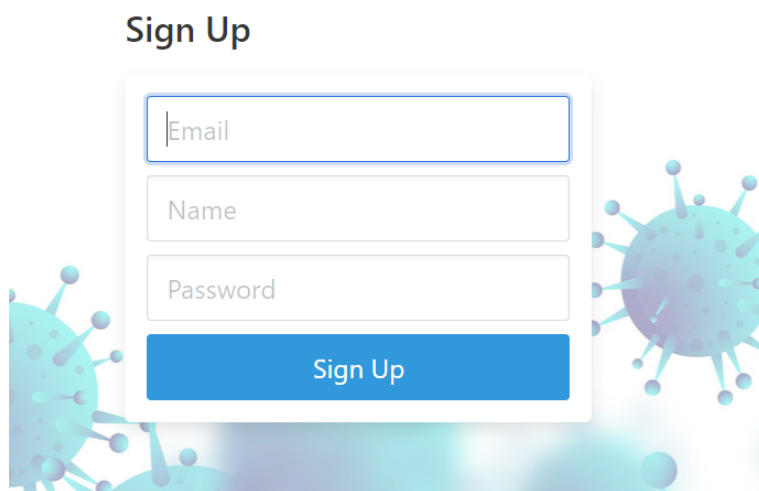
La figure ci-dessus, représente la page d'accueil de notre application :

- Sur cette page nous avons un tableau qui contient les dernières données Covid-19 des régions françaises.
- Le visiteur du site a la possibilité de passer sur la page suivante pour consulter les données des régions manquantes sur la première page.
- Le visiteur a la possibilité d'effectuer une recherche par date.

Comme nous avons déjà cité sur notre schéma fonctionnel, un visiteur a le droit de créer un compte.

Une fois que le compte utilisateur est créé, il sera automatiquement sauvegardé dans la BDD.

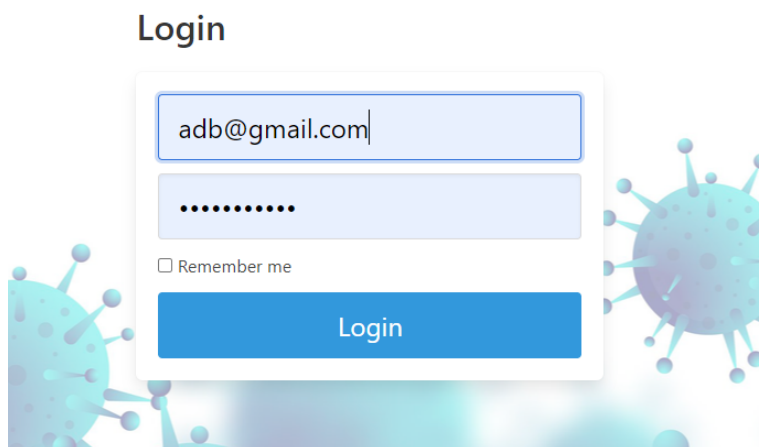
Nous avons crypté le mot de passe de l'utilisateur grâce à la librairie "werkzeug.security".  
Voici le formulaire d'inscription .



The image shows a 'Sign Up' form with a light blue background featuring a virus-like illustration. The form is a white box with a blue border. It contains three input fields: 'Email', 'Name', and 'Password'. Below these fields is a blue button with the text 'Sign Up' in white. The 'Email' field has a cursor at the end of the text.

FIGURE 18 – Création de compte

L'utilisateur peut accéder à son compte en saisissant son mail et mot de passe.  
Voici le formulaire de connexion.



The image shows a 'Login' form with a light blue background featuring a virus-like illustration. The form is a white box with a blue border. It contains two input fields: the first contains 'adb@gmail.com' and the second contains a series of dots. Below these fields is a checkbox labeled 'Remember me' and a blue button with the text 'Login' in white. The first input field has a cursor at the end of the text.

FIGURE 19 – Connexion

Pour la Gestion d'utilisateur, nous avons utilisé la librairie Flask\_admin, cette dernière nous a permis d'avoir une interface administrateur.


CovidApp Home User				
List (3) Create With selected ▼				
<input type="checkbox"/>		Email	Password	Name
<input type="checkbox"/>	 	lamia.kaclaissa@gmail.com	sha256\$Y3CU3bEu\$8c8f4bd83f6fdaa7620d0b4eb1141328ee6d7a6ad54b87a815cbced5a188094	Lamia Adjir
<input type="checkbox"/>	 	adjiriamia@gmail.com	sha256\$dhieiml6I\$8c5cf7fca3d0d19b998d7d543721078b3586f231d917281ff9f3cecadfbffce3	adjiriamia@gmail.com
<input type="checkbox"/>	 	adb@gmail.com	sha256\$Vjplml4F\$8214f56a8e90cd500b928c3df8bceb83dc56c3d2fdc4605764d64cc2e3ad4b84	Lamoia

FIGURE 20 – Interface Administrateur

Depuis cette interface l'administrateur peut Créer, modifier et supprimer des utilisateurs.

CovidApp Home User	
List	Create
Roles	<input type="text"/>
Email	<input type="text"/>
Password	<input type="password"/>
Name	<input type="text"/>
<input type="button" value="Save"/> <input type="button" value="Save and Add Another"/> <input type="button" value="Save and Continue Editing"/> <input type="button" value="Cancel"/>	

FIGURE 21 – Interface Administrateur- Nouveau Compte

## 0.2 Présentation du Dashboard final

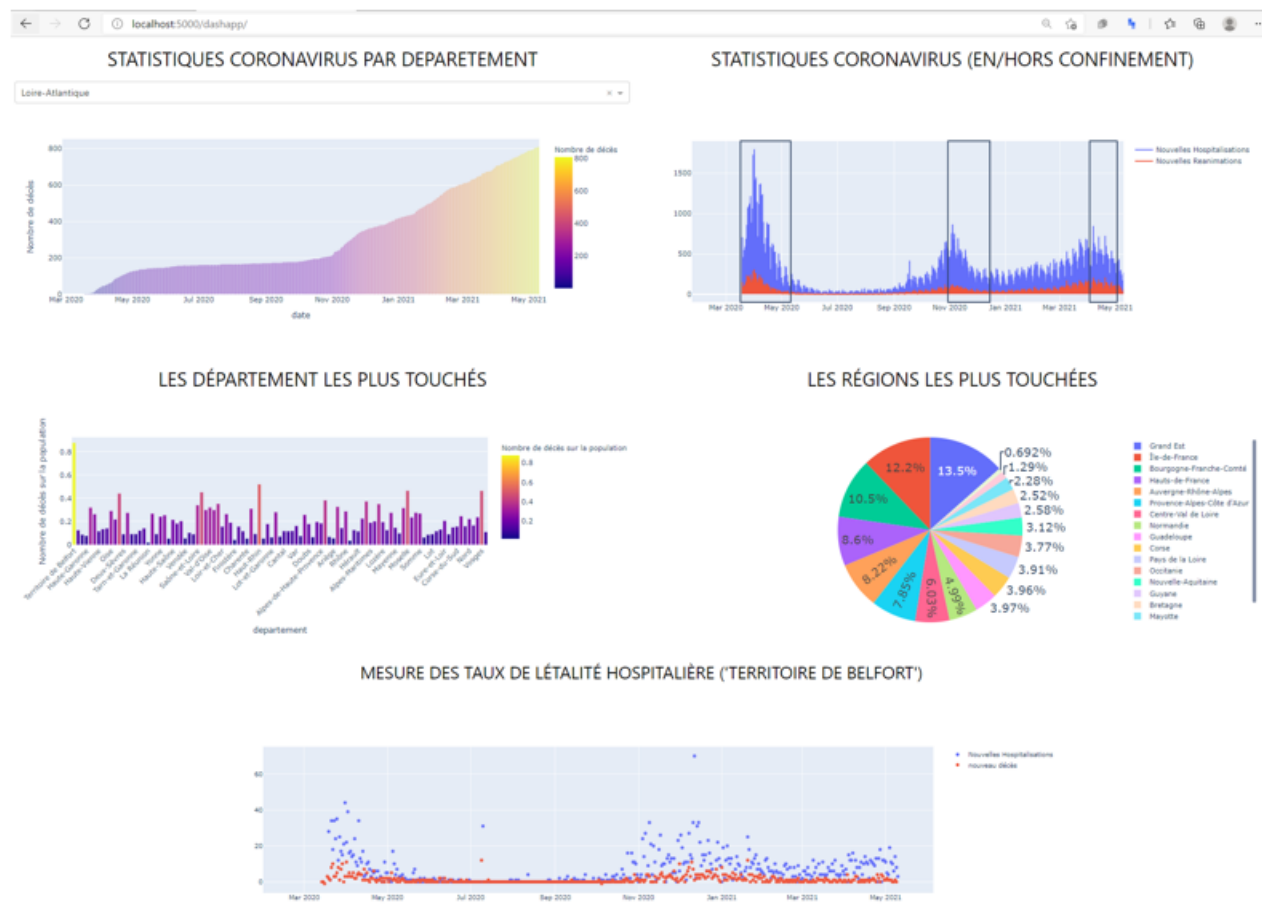


FIGURE 22 – Dashboard Final.

### 1. Statistique Coronavirus par département :

Un premier graphe dynamique a été réalisé pour afficher les statistiques du coronavirus par département.

L'utilisateur a l'option de sectionner le département souhaité pour afficher le résultat correspondant.

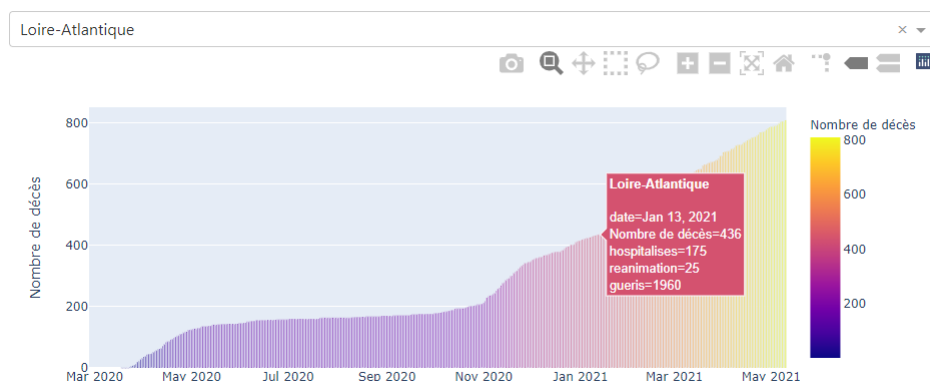


FIGURE 23 – Statistique CoronaVirus par département.

Ce graphe représente le nombre de décès pour chaque département depuis le début de la pandémie, les autres données(hospitalises, réanimation, guéris..) sont affichées en passant la souris sur le graphe.

### 2. Statistique CoronaVirus pendant et hors confinement :

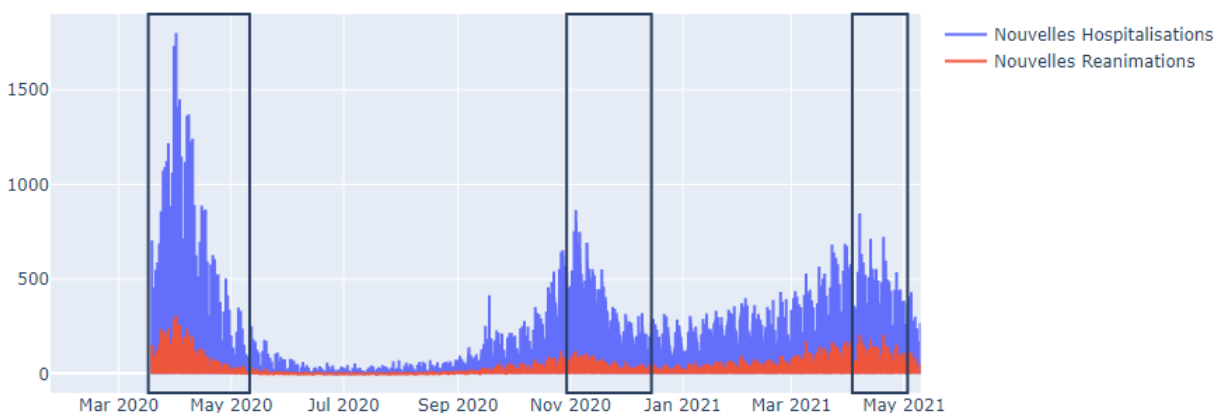


FIGURE 24 – Statistique CoronaVirus pendant et hors confinement

Un deuxième graphe a été réalisé pour illustrer l'impacte des trois confinements sur le nombre des hospitalisations et de patients en réanimation.

Les périodes concernées sont encadrées. Nous pouvons remarquer une baisse importante sur les cas recensés durant les confinements.

### 3. Les départements les plus touchés en terme de Décès :

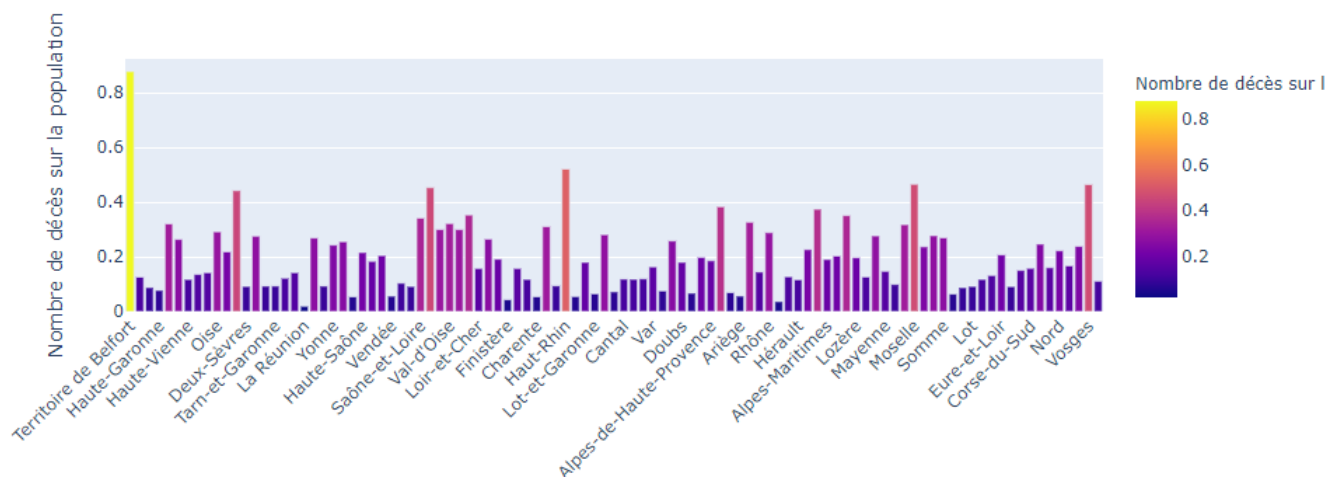


FIGURE 25 – Les départements les plus touchés(Décès)

Pour avoir des résultats fiables et cohérents, Il fallait calculer, pour chaque département, le rapport entre le nombre de décès et le nombre d'habitants. Cependant, comme nous n'avions pas la dernière information, nous avons été obligée de mettre à jour la base de données pour l'ajouter.

A la date du 11-05-2021, ce graphe montre que "Territoire de Belfort" est le département plus touché en nombre de décès.(565 décès /138589 habitants).

#### 4. Les régions les plus touchées en terme de Décès :

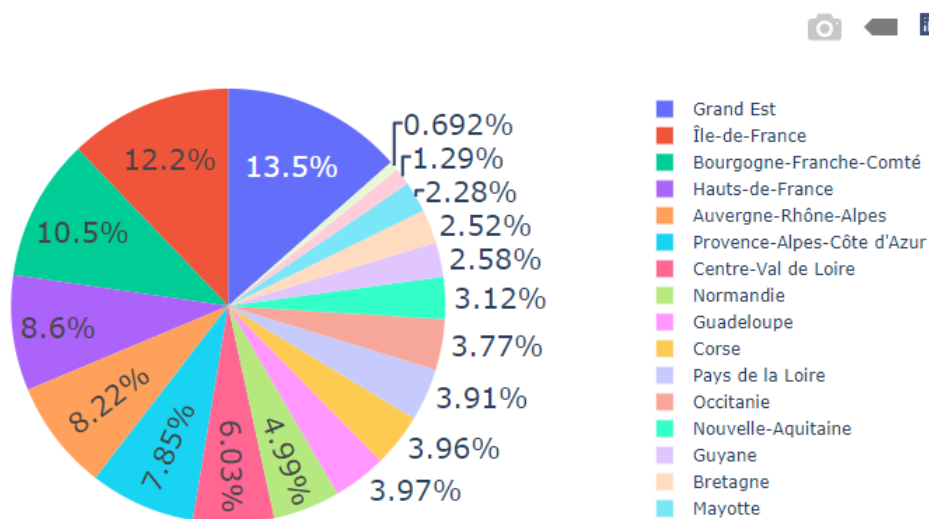


FIGURE 26 – Les régions les plus touchées(Décès)

à la date du 11-05-2021, Ce graphe montre que "Grand Est" est la région le plus touchée en nombre de décès.( 8892 décès /5522476 habitants).

### 5. *Mesure des taux de létalité hospitalière :*

Ce dernier graphe compare le nombre de morts par rapport au nombre de patients hospitalisés, dont le but est de mesurer la performance des systèmes de soins.

nous avons choisi de faire cette étude sur le département le plus touché à savoir "Territoire de Belfort".

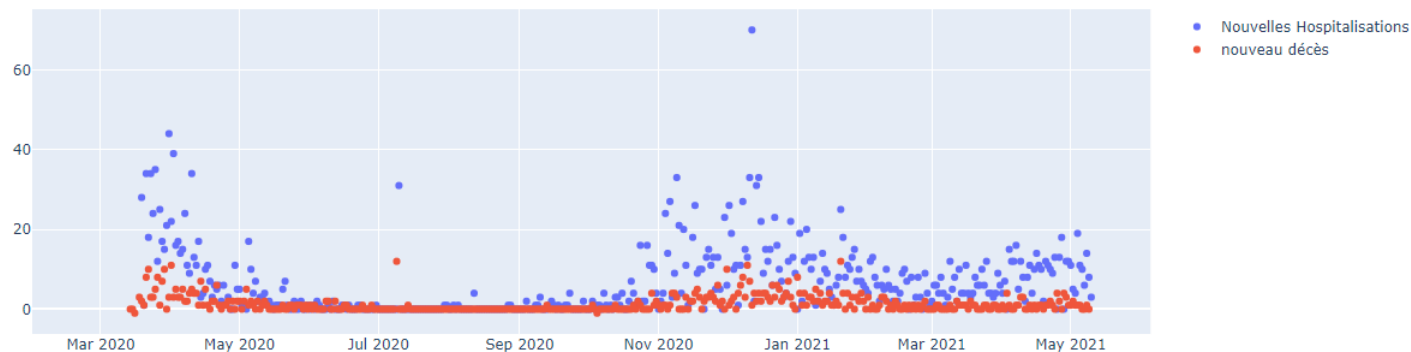


FIGURE 27 – Mesure des taux de létalité hospitalière

sachant que notre BDD contient le nombre de décès cumulé depuis le début de la pandémie ; nous avons été amenée à calculer le nombre des nouveaux décès pour le comparer aux nouvelles hospitalisations. pour cela voici la requête SQL réalisée :



```

select
  c.date,
  c."nouvellesHospitalisations",
  c.departement,
  c.deces,
  lag(deces) over (
    order by date ) as previous_deces,
  lag(deces) over (
    order by date desc) - deces as nouveau_deces
from
  "CleanDataByDepartement" c
where
  departement = 'Territoire de Belfort'
order by date desc

```

DataByDepartement

: c.date, c."nouvellesHospitalisations", c.departement, c.deces, lag(deces) ove

date	123 nouvellesHospitalisations	ABC departement	123 deces	123 previous_deces	123 nouveau_deces
2021-05-10	3	Territoire de Belfort	565	565	[NULL]
2021-05-09	8	Territoire de Belfort	565	564	0
2021-05-08	14	Territoire de Belfort	564	564	1
2021-05-07	6	Territoire de Belfort	564	564	0
2021-05-06	10	Territoire de Belfort	564	563	0
2021-05-05	11	Territoire de Belfort	563	562	1
2021-05-04	19	Territoire de Belfort	562	561	1
2021-05-03	4	Territoire de Belfort	561	559	1
2021-05-02	5	Territoire de Belfort	559	558	2
2021-05-01	11	Territoire de Belfort	558	557	1
2021-04-30	12	Territoire de Belfort	557	554	1
2021-04-29	12	Territoire de Belfort	554	550	3
2021-04-28	0	Territoire de Belfort	550	550	4

nous remarquons que le nombre d'hospitalisation est toujours supérieur au nombre de décès.

## Quatrième partie

### Conclusion générale et perspectives

## 0.3 Conclusion générale et perspectives

L'objectif de ce projet est de réaliser un Dashboard de visualisation graphique pour suivre l'évolution de la Covid-19 en France et l'impact des mesures sanitaires mises en place depuis le début de cette pandémie.

Dans la réalisation de ce projet nous nous sommes appuyée sur des données mises à disposition par des organismes spécialisés. Plusieurs étapes de collect, restructuration et stockages des données ont été appliquées pour atteindre les résultats souhaités.

Nous estimons pour notre part que l'objectif assigné à ce projet a été globalement atteint et avons constaté que les outils de visualisation permet une meilleure visibilité, d'ailleurs nous avons conclu dans une certaine mesure que le confinement a apporté un soulagement important sur les services d'hospitalisation et réanimations grâce à la baisse remarquée sur le graphe.

D'une manière générale, la réalisation de ce travail n'a pas été facile notamment sur le plan technique, l'utilisation de l'ORM sqlalchemy était une première, ce qui a nécessité un certain temps d'apprentissage et d'adaptation en programmation orienté objet, Cependant, grâce aux efforts consentis notamment le test de plusieurs solutions nous nous sommes enfin aboutie à ce résultat.

En perspective, de nombreuses améliorations peuvent être envisagées pour le travail que nous avons accompli afin d'obtenir une exploitation plus poussée de notre outil tel que :

- Le croisement de notre base de données avec une autre de vaccination contre la covid-19.
- La Proposition de nouvelles fonctionnalités sur la base d'un schéma de données continuellement enrichi grâce à la mise à jour des données.

Innovant, Utile, Répondant à un réel besoin, ce système sera très certainement d'une aide des plus précieuses pour toute organisation en quête de l'évolution de cette pandémie.

## 0.4 Références

[1] : <https://www.inserm.fr/information-en-sante/c-est-quoi/c-est-quoi-coronavirus>

[2] : [https://fr.wikipedia.org/wiki/Confinements\\_li%C3%A9s\\_%C3%A0\\_la\\_pand%C3%A9mie\\_de\\_Covid-19\\_en\\_France](https://fr.wikipedia.org/wiki/Confinements_li%C3%A9s_%C3%A0_la_pand%C3%A9mie_de_Covid-19_en_France)

[3] : <https://flask-login.readthedocs.io/en/latest/> [4] : <https://dash.plotly.com/introduction>