
SITE COVI-ITALIE

Projet Intégré

Amran ABDOURAZAK ABDILLAHI
Pauline ATTAL
Christelle KIEMDE
Véronique HOUNDONOUGBO
Prince MEZUI

M1 Informatique
2021 – 2022

SOMMAIRE

<i>Sommaire</i>	1
<i>Introduction</i>	2
<i>Objectif du projet</i>	3
<i>Choix des du thème</i>	3
<i>Choix des réseaux pour la récolte des données</i>	4
<i>Choix des technologies et langages</i>	4
<i>Environnement de travail</i>	4
<i>Selenium WebDriver</i>	5
<i>Dash</i>	5
<i>Flask</i>	5
<i>Méthodologie agile</i>	6
<i>Définition des backlogs du projet</i>	6
<i>Les technologies associées a la methodologie</i>	7
<i>Déroulement technique du projet</i>	8
<i>La collecte des données</i>	8
<i>La préparation des données</i>	10
<i>L'analyse des données</i>	12
<i>Création des graphiques</i>	13
<i>Réalisation de l'interface web</i>	14
<i>Méthodologie adoptée</i>	14
<i>Environnement de travail</i>	14
<i>Outils de développement</i>	15
<i>Résultats</i>	18
<i>Maquettage</i>	18
<i>Présentation des résultats</i>	19
<i>Conclusion</i>	20
<i>Sitographie</i>	21
<i>Annexes</i>	22

INTRODUCTION

Dans le cadre des cours de Gestion de Projet et de Projet Intégré, du Master 1 Informatique, nous étions amenés à créer une interface présentant des données en lien avec le tourisme dans les sites recensés par l'UNESCO. Nous avons décidé de créer un site web pour présenter l'analyse de nos données collectées. Nous avons donc créé un site appelé Covi-Talie, qui permet de voir l'influence du Covid-19 sur le tourisme italien.

Notre site a pour but de présenter des analyses de flux touristique en Italie pendant la période de covid, entre et dans les monuments les plus connus, grâce à des graphiques visuels et interactifs pour les visiteurs du site web. Nous permettons aussi la visualisation de graphiques qui prédisent le comportement touristique en fonction des événements futurs liés au Covid-19.

Le contenu du site sera décrit dans une partie dédiée de ce rapport : *Résultat*. Un lien sera disponible pour visiter le site, visualiser les graphiques et consulter leur analyse.

Durant le projet, nous avons utilisé la méthodologie d'organisation SCRUM. Pour certains membres du groupe, c'était le premier projet réalisé avec cette méthode de travail. Nous allons rapidement décrire dans une sous-partie de ce rapport, comment nous avons appliqué cette méthodologie, qui a articulé notre organisation tout au long de la réalisation du projet.

Les étapes clés de notre projet ont été :

1. la collecte des données,
2. le nettoyage des données,
3. l'analyse des données,
4. la création des graphiques liées aux analyses,
5. la création de l'interface pour présenter les graphiques et analyses.

Tout ce processus sera décrit dans différentes parties de ce rapport.

Afin de bien organiser les tâches dans ce projet, en plus de l'application de la méthode agile, nous avons décidé d'utiliser Google Drive pour partager instantanément tous les documents en lien avec le projet : scripts python sous Google Collab, scripts R, fichiers de données, documents textes. Ces fichiers nous ont permis d'avoir une trace de tout l'avancement du projet et nous ont aidés à la rédaction du rapport final. Nous avons aussi mis les 2 fichiers importants correspondant à celui du Scrum Master et celui du Product Owner. Le fichier du Scrum Master résume tout notre avancement des sprints et celui du Product Owner contient notre travail technique accompli et nos difficultés rencontrées. Ces deux fichiers étaient présentés chaque semaine au professeur de gestion de projet, qui représentait pour nous le client.

Durant notre projet, nous avons fait certaines tâches en groupe et certaines tâches individuellement. Les tâches de réflexions et de formations à des techniques étaient communes et pour les tâches plus précises, nous travaillons individuellement dans le respect de la méthode agile. Nous avons organisé de nombreuses réunions et temps de travail en dehors des Travaux Pratiques.

Ainsi, nous avons réussi à avancer et finir notre projet avec les objectifs que nous nous étions fixés.

OBJECTIF DU PROJET

CHOIX DES DU THÈME

Notre projet s'articule autour du traitement des flux touristiques italien. Notre objectif est d'analyser l'impact du covid sur le tourisme italien. Le but de cette analyse autour du covid est d'aider les italiens à comprendre et se rendre compte de l'impact qu'a eu le covid pour le tourisme de leur pays. Pour ce faire, il va falloir axer notre réflexion sur les dates pour analyser les flux avant le covid et pendant le covid. Dans cette partie nous allons détailler les choix que nous avons faits pour réaliser cette problématique.

La première étape pour nous, fut de définir plus précisément le sujet du projet, en choisissant des sites de l'UNESCO que nous devrons analyser. Nous avons donc fait un brainstorming pour recenser les idées de chacun, et nous avons vite été tous en accord pour converger vers l'Italie, comme dit plus haut. Nous avons remarqué de nombreux sites historiques, ainsi que naturels et culturels. Nous avons donc pensé que ce lieu pouvait réunir beaucoup de touristes européens et mondiaux et qu'il pouvait être intéressant de choisir des sites UNESCO de ce pays.

La seconde étape était donc de se renseigner sur les principaux sites unesco de l'Italie pour n'en retenir que 2. Nous nous sommes donc répartis ce travail de recherche, pour ensuite se mettre d'accord sur les deux sites suivants :

1. Site historique de Rome
2. Venise et sa lagune



Ces deux sites étant tous les deux situés en Italie à une distance non loin, nous pensons donc qu'une analyse inter-site serait donc intéressante.

Pour ces deux sites, afin d'effectuer une analyse des flux *dans* le site, nous avons décidé de 3 monuments phares pour chaque site touristique, qui recense le plus d'avis et de postes sur les sites internet tels que Google maps, TripAdvisor ou encore Instagram.

Nous avons donc convenu :

- Le Panthéon, le Colisée et la fontaine de Trevi pour Rome.
- La place Saint-Marc, le palais de Doges et le pont du Rialto pour Venise.

Pour le moment, seuls 2 sites touristiques UNESCO et au total 6 monuments ont été analysés. Cependant l'objectif d'une telle analyse axée sur l'Italie est de réaliser le travail sur un plus large panel de sites touristiques et de monuments. Toutefois pour le délai imposé, il était donc important de délimiter le projet comme nous l'avons fait.

CHOIX DES RÉSEAUX POUR LA RÉCOLTE DES DONNÉES

La technique de récolte d'information sur le web, en informatique, s'appelle le *Scraping*.

Nous avons donc dû choisir des sites web répertoriant des avis sur des lieux touristiques. La stratégie était de choisir des sites mondialement connus pour avoir des avis d'une grande étendue. Le site internet Tripadvisor est le premier site qui nous a paru évident sur lequel on pourrait récolter un grand nombre d'informations. Nous avons aussi intuitivement pensé aux réseaux sociaux Instagram et Facebook. Par la suite nous avons aussi eu l'idée de récolter les informations sur le site Google Maps. Nous avons donc testé le scraping avec Facebook, Instagram, TripAdvisor et Google, mais ce sont seulement ces deux derniers pour lesquels nous avons pu récolter des données traitables.

Tripadvisor		
Colisée  147 149 •	Colisée Colosseo 4,7 ★★★★☆ 274 694 avis	

CHOIX DES TECHNOLOGIES ET LANGAGES

Pour le langage de programmation nous avons tout de suite pensé à python car nous savons qu'il était possible de faire de la récolte de données facilement grâce à des librairies de scraping, qu'il est aussi facile de manipuler des données avec les Data Frame de la librairie Pandas, et qu'il est aussi possible de faire du web avec le Framework Flask, et des dashboards avec le Framework Dash.



ENVIRONNEMENT DE TRAVAIL

Nous avons choisi Google Colab pour l'écriture de nos scripts de scraping et de nettoyage de ces données. Cela nous permettait de partager facilement nos scripts dans le drive que nous avions créé au début du projet.



Par la suite, pour la création de l'interface web, nous avons travaillé avec la distribution anaconda. Cela nous permettait d'écrire nos codes dans l'environnement de travail spyder, tout en ayant une gestion correcte des librairies que nous utilisons. Pour partager nos dossiers et fichiers, nous avons utilisé git. Nous décrirons cela un peu plus bas dans le rapport.



SELENIUM WEBDRIVER

Un temps important de réflexion et de réalisation a été consacré pour la récolte des données. Cette compétence n'était maîtrisée par aucun membre de groupe, il a donc fallu faire des recherches sur les librairies de scrapping. Celle qui a été retenue fut sélénum. Nous avons donc dû nous former à cette librairie et écrire les scripts spécifiques à chaque réseau social. Cette étape sera décrite dans la section *récolte des données*.

Sélénum WebDriver est un framework qui permet la simulation d'interactions avec les utilisateurs dans n'importe quel navigateur. Nous nous en sommes servi pour simuler l'ouverture de page web, et récupérer le contenu texte des balises HTML de chaque page web ouverte.



DASH

Les graphiques de notre site doivent être organisés dans des tableaux de bord. Un tableau de bord regroupe des informations visuelles (graphiques) d'un même sujet et sur une seule page. La création de ces tableaux de bord est possible avec le framework de python Dash qui intègre la librairie Plotly pour la création des graphiques . Nous détaillerons son utilisation de la section *Création des graphiques*.



FLASK

Pour la réalisation de l'interface Web, nous avons choisi le micro-framework Flask de python, qui permet la création d'une application Web en très peu de temps et facilement. Son fonctionnement sera expliqué dans la partie *réalisation de l'interface web*.



MÉTHODOLOGIE AGILE

Durant ce projet, nous avons organisé la gestion du projet sous les règles de la méthodologie agile. Dans cette méthode de travail, le client est intégré au cœur du projet. Cela permet d'avoir un avis régulier sur l'avancement du projet. La méthode agile prône donc l'adaptation des procédés de création au fil de l'évolution du projet en fonction des demandes du client, peu importe l'avancement du projet. Le projet est décomposé en étapes de développement. Des objectifs sont fixés à court terme. On appelle cela des sprints.

Deux rôles clés entrent en jeu dans la méthode agile : le Scrum Master et le Product Owner.

Le **Scrum Master** doit s'assurer du bon déroulement du projet, c'est-à-dire dans le respect de la méthode agile. La méthode agile sous-entend d'effectuer des tâches s'enchaînant les unes à la suite des autres. Un ensemble de tâches fait l'objet d'un sprint qui se réalise pendant une durée définie. L'estimation du temps et le respect de cette estimation pour le sprint et pour chaque tâche du sprint sont des notions très importantes dans cette façon de travailler. Le scrum master est chargé donc de faire respecter ces règles de méthodologies.

De plus, le **Product Owner** joue aussi un rôle majeur lors d'un projet réalisé avec cette méthode. En effet, il est chargé de faire la communication entre son équipe et le client pour lequel l'équipe travaille. Le product owner montre l'avancement du projet à chaque fin de sprint, il demande l'avis au client, qui lui aussi joue un rôle important dans le projet. Ces échanges entre le client et l'équipe de développement permettent de se rendre compte si le projet avance correctement dans les attentes du client, et dans le respect des dates de livraison.

DÉFINITION DES BACKLOGS DU PROJET

Pour démarrer le projet dans les règles de la méthodologie agile, il a fallu que nous définissions les backlogs du projet. Les backlogs constituent une liste de fonctionnalités qui sont utiles à la conception de notre site web.

 FU-27	Definition du projet et choix
 FU-28	Formations aux technologies choisies
 FU-23	Extraction des données sur le web
 FU-25	Nettoyer et pré-traitement des fichiers générés
 FU-30	Analyse statistique sur les données
 FU-31	Création des graphes basées sur les analyses statistiques
 FU-32	Création de l'interface web
 FU-33	Prévision des données

Ces backlogs sont ensuite décomposés en items, puis ces items sont placés dans des sprints. Un sprint est composé de plusieurs tâches utiles à la réalisation de l'item contenu dans le sprint.

LES TECHNOLOGIES ASSOCIÉES A LA METHODOLOGIE

Afin d'être dans les règles de la méthode agile, nous avons utilisé l'application Jira qui permet une collaboration et une organisation de projet adapté à la méthode agile. Nous avons pu définir nos backlogs, nos items ainsi que nos sprints grâce à Jira. Chaque membre du groupe avait accès au compte Jira, afin de visualiser l'ensemble des tâches et leur avancement par les coéquipiers.



L'utilisation de Git a aussi été utile lors du développement de l'interface web. Nous avons travaillé avec l'interface git GitHub. Git permet l'enregistrement de différents fichiers tout au long des différentes étapes d'un projet. Cela nous permet de garder une trace de ce qui a été fait et de revenir à une phase précédente d'un fichier.



La communication était faite à travers un groupe de conversation whatsapp. En plus de nos échanges virtuels, nous avons organisé de nombreuses réunions externes au cours pour l'organisation du projet et pour travailler ensemble et échanger plus facilement.



DÉROULEMENT TECHNIQUE DU PROJET

Dans cette partie, nous allons détailler et expliquer notre déroulement du projet d'un point de vue technique. Nous avons décidé d'utiliser le langage Python pour toutes les étapes techniques du projet, qui sont :

1. la collecte des données,
2. la préparation des données,
3. l'analyse des données,
4. la création des graphiques,
5. la création du site web.

Cependant, pour l'analyse, nous avons aussi utilisé le logiciel R en amont, qui est très bien adapté pour les analyses de séries temporelles.

LA COLLECTE DES DONNÉES

Comme dit précédemment, nous avons récolté des données grâce à l'API de Python *Selenium*. Nous avons finalement choisi les réseaux **TripAdvisor** et **GoogleMap**, pour les monuments *Panthéon*, *fontaine de Trevi* et *le Colisée* faisant partie du site UNESCO de Rome, et les monuments *Palais des Doges*, *place Saint Marc* et *le pont du Rialto* pour le site UNESCO de Venise.

Dans cette partie, nous allons présenter et expliquer l'API Python *Sélénum* ainsi que nos scripts Python qui nous ont servi pour la récolte des données.

Tout d'abord, *Sélénum* est un outil permettant l'automatisation de tests sur le Web. Cependant, nous nous sommes servi de *Sélénum* afin de créer un automate qui navigue dans des pages Web comme le ferait un vrai utilisateur. Une fois que notre script nous a conduits au bon endroit de la page Web, il charge les données statiques, visibles par le navigateur, de la page HTML. Notre travail ensuite est de cibler les bonnes balises HTML via un chemin XPATH, pour charger seulement le texte qui nous intéresse. XPath est un langage de requête qui permet de localiser une portion d'un document HTML. De plus, chaque page est structurée différemment. La page web Google Maps n'est pas la même que celle de TripAdvisor, nous n'allons donc pas chercher les données au même endroit. Nous spécifions donc des chemins XPATH différents.

Pour cela, Selenium nous fournit plusieurs fonctions qui nous ont servi pour la création de nos scripts et que nous allons décrire brièvement ci-dessous.

Il est important de noter qu'un script est spécifique à une page Web. En effet, nous demandons au pilote (Chrome pour nous) d'ouvrir une URL, et une fois sur ce lien, un enchaînement d'opérations doit être réalisé par le visiteur en fonction de la structure du site : connexion, acceptation de cookies, tri des données de la page, recherche à faire, etc.

```
driver = webdriver.Chrome("/chemin/chromedriver") # Driver Chrome
URL = "https://www.tripadvisor.fr"
driver.get(URL) # Récupération de la page web
```

Le code permettant de cliquer sur un bouton est le suivant :

```
bouton = driver.find_element_by_xpath('XPATH')
bouton.click()
```

A chaque lancement du script, un fichier CSV se crée et s'enregistre sur votre ordinateur. Ce fichier contient donc toutes les données lues sur le web. Celles-ci sont organisées dans différentes colonnes. Voici les informations concernant le fichier regroupant la totalité des données récoltées. Nous avons 168285 lignes et 8 colonnes.

```
df = pd.read_csv("database.csv", sep=";")
print(df.info())

RangeIndex: 168385 entries,
Data columns (total 8 columns):
 #   Column
 --- 
 0   Pseudo      #identifiant de la personne qui a écrit le commentaire
 1   Ville        #ville d'origine de la personne
 2   Pays         #Pays d'origine de la personne
 3   Date         #Date à laquelle la personne a posté le commentaire
 4   Titre        #Titre du commentaire
 5   Commentaire  #commentaire
 6   Label         #monument pour lequel le commentaire a été laissé
 7   Langue       #langue dans laquelle le commentaire a été écrit
```

Afin que nos données soient représentatives, nous avons décidé de récupérer des commentaires dans plusieurs langues pour avoir une vision plus globale de l'impact du tourisme en Italie. Le site Google Maps regroupe tous les commentaires peu importe la langue, le script de récolte de données n'a donc été lancé qu'une seule fois pour chaque site touristique. Cependant, pour TripAdvisor, il a fallu sélectionner un filtre sur la langue. Nous avons donc lancé plusieurs fois le script, puis nous avons assemblé tous les fichiers CSV que chaque lancement de script nous construisait. Voici les 5 langues que nous avons retenues : ['francais' 'espagnol' 'italien' 'anglais' 'portugais'].

Nous avons voulu collecter également les données sur Facebook et Instagram. Cependant, nous avons rencontré des problèmes, en effet nous nous sommes retrouvés avec des comptes bloqués après avoir pu récupérer une quantité minime de données sur ces 2 sites. Par la suite nous avons décidé de prendre juste les données de Tripadvisor et Google Maps.

Pour la récolte des données nous avons rencontré un problème pour les sources que nous voulions récolter. En effet nous voulions réaliser des scripts qui récupèrent des données sur les réseaux de Facebook et Instagram mais cela s'est avéré plus compliqué que prévu. Le site instagrame n'est pas facilement navigable avec la technique que nous avons utilisée avec la librairie Selenium. De plus, pour le site Facebook, les posts existant sur les différents monuments ne sont pas forcément postés par des personnes qui ont visité récemment. Cela nous aurait donc donné des données non conformes et nous avons préféré garder seulement TripAdvisor et Google Maps qui nous fournissait un nombre de données qui nous satisfaisait.

LA PRÉPARATION DES DONNÉES

Afin de pouvoir traiter nos données pour les analyses, il a fallu les nettoyer. Le nettoyage consiste à détecter et de corriger ou supprimer les erreurs.

La principale erreur à corriger pour nous était les dates. En effet, le format des dates était en chaîne de caractères pour les données provenant de TripAdvisor.

Données non transformées TripAdvisor :

```
[gen 2021 , feb 2021 , mar 2021 , apr 2021 , mag 2021 , giu 2021 , lug 2021 , ago 2021 , set 2021]
```

Données transformées TripAdvisor :

```
[2021-01-01, 2021-02-01, 2021-03-01, 2021-04-01, 2020-05-01, 2020-06-01, 2021-07-01,  
2020-08-01, 2020-09-01]
```

Du côté de Google Maps, la transformation était un peu plus compliquée. En effet, le format de la date récoltée était aussi en chaîne de caractères, mais la formulation était plus compliquée à transformer. De plus, la date devait être construite à partir de la date du jour de l'extraction des données. Nous avons donc utilisé une autre librairie pour soustraire de la date du jour, une certaine quantité de jours, mois, années etc.

```
from dateutil.relativedelta import *  
date((datetime.now()-relativedelta(days=num)).year, \  
     (datetime.now()-relativedelta(days=num)).month, \  
     (datetime.now()-relativedelta(days=num)).day))  
#days peut être remplacée par months, weeks, etc.  
#num contient la quantité de jours, mois, années etc. passées
```

Données non transformées Google Maps :

```
[il y a 4 semaines , il y a 3 ans , il y a un mois]
```

Données transformées Google Maps :

```
[2021-10-08, 2018-10-01, 2021-09-01]
```

La fonction ci-dessous nous a permis de faire la transformation pour les dates provenant de la récolte de données du site TripAdvisor. Celle-ci va parcourir notre champ Date de type chaîne de caractère, pour le transformer en champ “Date” de type Date. Nous avions à décomposer le champ de base en 2 en identifiant l'année (deuxième mot de la chaîne de caractères) et le mois (premier mot de la chaîne de caractères).

```
def changeDate(data):  
    for i in range(len(data)):  
        #si la date est au bon format, on ne change rien  
        if (type(data.Date[i]) == datetime):  
            data.Date[i] = data.Date[i].date()  
        elif (type(data.Date[i]) == str):  
            #recuperation de l'année  
            an = data.Date[i].split()[1] #deuxième mot  
            #recuperation du mois  
            mois = data.Date[i].split()[0] #premier mot
```

```

#attribution du numéro de mois en fonction de la chaine de caractere
if (mois == 'gen'):
    ms = 1
elif (mois == 'feb'):
    ms = 2
elif (mois == 'mar'):
    ms = 3
elif (mois == 'apr'):
    ms = 4
elif (mois == 'mag'):
    ms = 5
elif (mois == 'giu'):
    ms = 6
elif (mois == 'lug'):
    ms = 7
elif (mois == 'ago'):
    ms = 8
elif (mois == 'set'):
    ms = 9
elif (mois == 'ott'):
    ms = 10
elif (mois == 'nov'):
    ms = 11
elif (mois == 'dic'):
    ms = 12
#création de la date avec la librairie Date
data.Date[i] = date(int(an), ms, 1)

```

Nous avons aussi essayé de mettre en forme le champ pays car nos données récoltées dans les autres langues étaient conservées dans la langue originale du poste. Le but était donc de traduire ce champ, à l'aide de la librairie googletrans de python. Cependant, nous avons des problèmes dus à la mauvaise traduction de la librairie, et aussi du au gros volume de données que nous imposons à la librairie.

Voici ci-dessous le code de cette transformation qui n'a donc pas pu être exécutée correctement.

```

from googletrans import Translator
trans = Translator()
translations = {}
for element in df["Pays"]:
    translations[element] = trans.translate(element).text
df.replace(translations, inplace = True)

```

L'analyse des données a consisté dans un premier temps à faire une description des données afin de mieux les appréhender. Ainsi nous avons d'abord étudié le flux touristique par site, puis par monument. Ensuite, nous avons fait une analyse de lien entre les deux sites. L'objectif était de voir si les touristes qui visitent Rome, visitent également Venise dans un intervalle de temps court(3 mois). Une autre analyse a été faite pour étudier l'évolution mensuelle du nombre de visites par monument et par site de 2017 à 2021. La seconde partie de l'analyse à consister à étudier le flux touristique au niveau des monuments avant et pendant les années de Covid. En effet, nous avons pensé que le Covid 19 aurait un impact sur le tourisme (les visites des sites touristiques) avec les confinements et les différentes fermetures de frontières. Allant dans ce sens, nous avons fait une comparaison des évolutions annuelles des flux touristiques par site avant et pendant les années de Covid.



La dernière partie de l'analyse à consister à faire une prévision sur 6 années en prenant 2023 comme année de fin de Covid. Nous avons d'abord transformé les données en séries temporelles, puis nous avons enlevé la tendance et la saisonnalité. Ensuite, nous avons utilisé ARIMA qui est un modèle statistique pour analyser ou prédire les données de séries temporelles. Nous avons pu donc construire plusieurs modèles avec les valeurs que l'autocorrélogramme nous a permis d'avoir. L'AIC a été utilisé comme critère de sélection du meilleure modèle qui servira à la prévision. Cette prévision nous permet de connaître le flux touristique sur les 6 années à venir et d'appréhender la situation de la pandémie du covid 19. Nous avons utilisé pour la première et deuxième partie de l'analyse le langage de programmation python. Pour la troisième partie,nous avons utilisé le langage de programmation et logiciel R qui est très bien adapté pour les analyses de séries temporelles.



Les résultats de l'analyse des données devraient être enregistrés dans un fichier csv afin de permettre la construction des graphes. Pour ce faire, nous avons utilisé la librairie csv en vue de stocker les différents résultats dans un fichier Excel d'extension csv.

L'étape de création des graphiques a été l'une des plus importantes car, durant la présentation de notre travail global, la qualité des visualisations sera très déterminante. Nous nous sommes donc entendus sur le fait de créer les graphiques les plus synthétiques mais également les plus interactifs possible afin de faciliter la compréhension. Pour ce faire, la librairie plotly.express (plus simplement notée px) nous à semblé être la plus adaptée. Grâce à elle, nous avons pu obtenir tous les graphiques qu'on désirait et la plus value est le rendu interactif des graphiques qui réagissent selon les mouvements de souris de l'utilisateur.



Comme le montre l'exemple ci-après, on voit qu'une fois le graphique créé, on peut le modifier à notre guise. Ici, on a créé un graphique en barre et on a pu ajouter des étiquettes qui donnent au-dessus de chaque barre, les valeurs mesurées. De plus, afin d'harmoniser le graphique avec le style de l'interface web, nous avons modifié les couleurs des barres et les couleurs de fond du graphique.

Une fois le graphique finalisé, on le transforme en JSON afin qu'il soit interprétable par Flask et qu'il puisse être incorporé dans l'interface web.

```
valeurs = []
#récupérer les valeurs contenues dans le curseur res
for row in res:
    valeurs.append(int(row[1]))
#créer un DataFrame manipulable, avec ces valeurs
data = pd.DataFrame({
    "Nombre de visites": valeurs,
    "Sites": ["Rome", "Venise"]
})
#création de la figure (fig) de type bar (histogramme)
fig = px.bar(data, x="Sites", y="Nombre de visites", text="Nombre de visites",
color="Sites", color_discrete_map=colors)
#amélioration des éléments d'informations visuel de la figure
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')
#amélioration de l'identité visuel (couleurs) de la figure
fig.update_layout(
    plot_bgcolor=colors['background'],
    paper_bgcolor=colors['background'],
    font_color=colors['text']
)
#conversion de la figure en objet JSON, interprétable par Flask
graphJSON = json.dumps(fig, cls=plotly.utils.PlotlyJSONEncoder)
```

RÉALISATION DE L'INTERFACE WEB

Nous avons décidé de créer un site internet afin de présenter nos graphiques de visualisation ainsi que ses interprétations. Nous avons mis en place un site internet *responsive design* pour être correctement visible sur tous les types de terminaux (PC, Smartphones, tablettes).

MÉTHODOLOGIE ADOPTÉE

Le patron **MVT** représente une architecture orientée autour de trois pôles : le **Modèle**, la **Vue** et le **Template**. Elle s'inspire de l'architecture MVC, très répandue dans les Framework web. Son objectif est de séparer les responsabilités de chaque pôle afin que chacun se concentre sur ses tâches.

Modèle

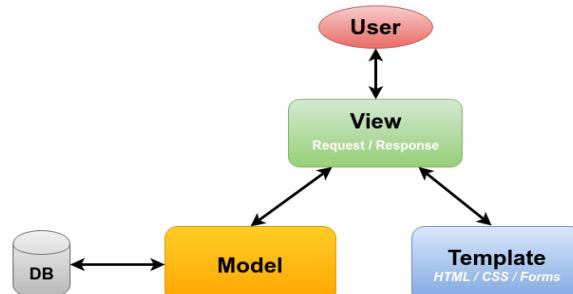
Le modèle interagit avec la base de données. Sa mission est de chercher dans une base de données les items correspondant à une requête et de renvoyer une réponse facilement exploitable par le programme. Dans notre cas c'est le fichier *model.py*

Template

Un template est un fichier HTML qui peut recevoir des objets Python et qui est lié à une vue. Il est placé dans le dossier **Template**.

Vue

La vue joue un rôle central dans un projet structuré en MVT : sa responsabilité est de recevoir une requête HTTP et d'y répondre de manière intelligible par le navigateur. Ce n'est d'autre que *view.py*



ENVIRONNEMENT DE TRAVAIL

Anaconda est une distribution libre et open source des langages de programmation Python particulièrement orientée pour des applications en data science. L'un des atouts majeurs d'Anaconda est sa simplification dans la gestion des packages et de leur dépendance. Nous avons créé un environnement de travail sous conda, afin qu'on puisse installer des packages Python.



Outils de développement

- **GIT** : est un logiciel de gestion de versions décentralisé. En plus d'être décentralisé, Git a été conçu pour répondre à trois objectifs : performances, sécurité et flexibilité. C'est donc un outil indispensable dans le développement de projet informatique.



- **GitHub** : est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. Voici le lien de notre dépôt :

https://github.com/amran-zak/Covi_talie_fr.git



- **PythonAnyWhere** : est un environnement de développement intégré en ligne et un service d'hébergement Web basé sur le langage de programmation Python. Nous avons utilisé PythonAnywhere pour mettre notre site web en ligne. Voici le lien du site :



Langages :

- **HTML5/CSS3** : sont les dernières versions des principaux langages Web validés par le World Wide Web Consortium (W3C). Complémentaire, le HTML permet d'afficher le contenu des pages web et le CSS vise à décrire la présentation de ces contenus. Les deux sont associés dans le cadre des développements pour le Web.



- **Javascript** : est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Avec les technologies HTML et CSS, JavaScript est parfois considéré comme l'une des technologies cœur du World Wide Web.



- **Python** : est un langage de programmation adapté à toutes sortes de projets car il est possible de faire du code fonctionnel, impératif ou orienté objet. Il est donc adapté pour toutes sortes de projets. De plus, Python offre de multiples librairies destinées à la data science.



Nous avons étendu les fonctionnalités de python au maximum pour la réalisation de notre projet en utilisant la librairie Flask qui permet de créer des sites web avec le langage python.

Voici un bout de code commenté pour montrer le fonctionnement de Flask.

```
#création de l'instance Flask dans le module courant
app = Flask(__name__)
#le décorateur app.route sert à déclarer une URL qui exécutera la fonction
ci-dessous, ici about()
@app.route('/about/')
def about():
    description = """Bienvenue sur le site Cove-Italie"""
    #fonction render_template() va appliquer la fonction about() au template
    passé en premier paramètre, ici about.html
    return render_template('about.html',
                           descriptionHTML = description)
    #Le paramètre descriptionHTML est spécifié dans le code du template, et
    prend la valeur contenue dans la variable description.

<!-- fichier about.html -->
<div>{{descriptionHTML}}</div>
```

- **SQL** : est un langage informatique normalisé servant à exploiter des bases de données relationnelles. Les requêtes SQL permettent de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles. Après avoir préparé les données que nous avons récoltées, nous les avons mises dans une base de données (SGBD) : PostgreSQL.



La mise en place d'une base de données nous permet de stocker le grand volume de données récoltées, sans avoir à les charger à chaque ouverture de la page web. Il est ensuite facile de récupérer les données à l'aide de requêtes SQL qui interrogent la base de données.

Voici le code résumé pour la connexion à la base de données et le chargement d'un fichier dans celle-ci. Et ensuite comment récupérer les données à l'aide d'une simple requête SQL.

```

conn = psycopg2.connect(
    user = "xxx",
    password = "xxx",
    host = "db-etu.univ-lyon2.fr",
    port = "5432",
    database = "xxx"
)
#ouverture de curseurs
cur_create = conn.cursor()
cur_select = conn.cursor()
#déclaration des requêtes
sql_create = '''CREATE TABLE NomTable(...);'''
sql_select = SELECT * FROM NomTable;'''
#exécution du curseur avec la requête
cur_create.execute(sql)
#chargement du fichier de données
df =pd.read_csv("fichier.csv", sep=",")
#insérer les données ligne par ligne
for i in range(0,len(df)):
    sql = """INSERT INTO NomTable (NomColonnes) VALUES (%s)"""
    value = (df[i][0])
    cur.execute(sql, value)
#exécution du curseur avec la requête
cur_select.execute(sql_select)

```

Bootstrap:

C'est une collection d'outils utiles à la création du design de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option.

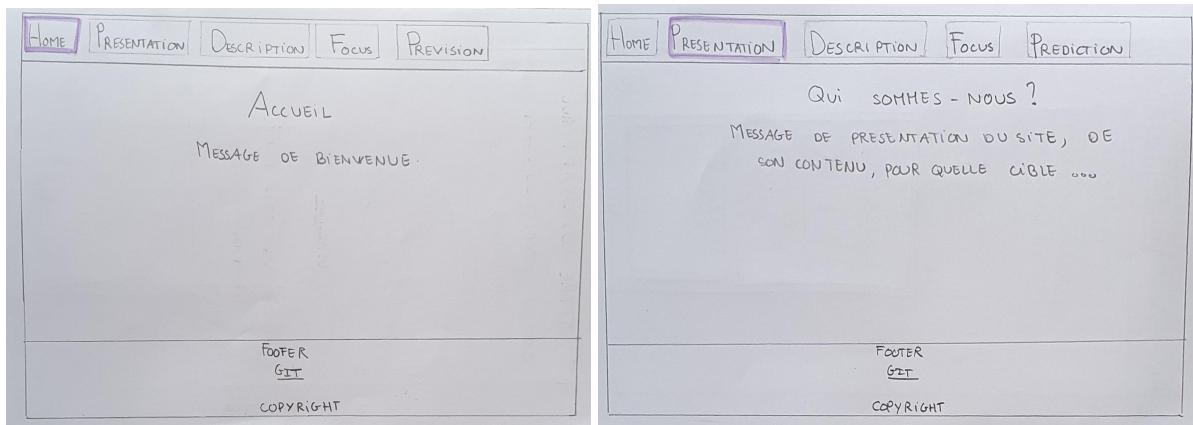


RÉSULTATS

La présentation des résultats se fait à travers l'interface web. Nous avons découpé nos analyses en trois grandes parties : une analyse descriptive, une analyse approfondie que nous avons appelée focus, et une analyse prédictive.

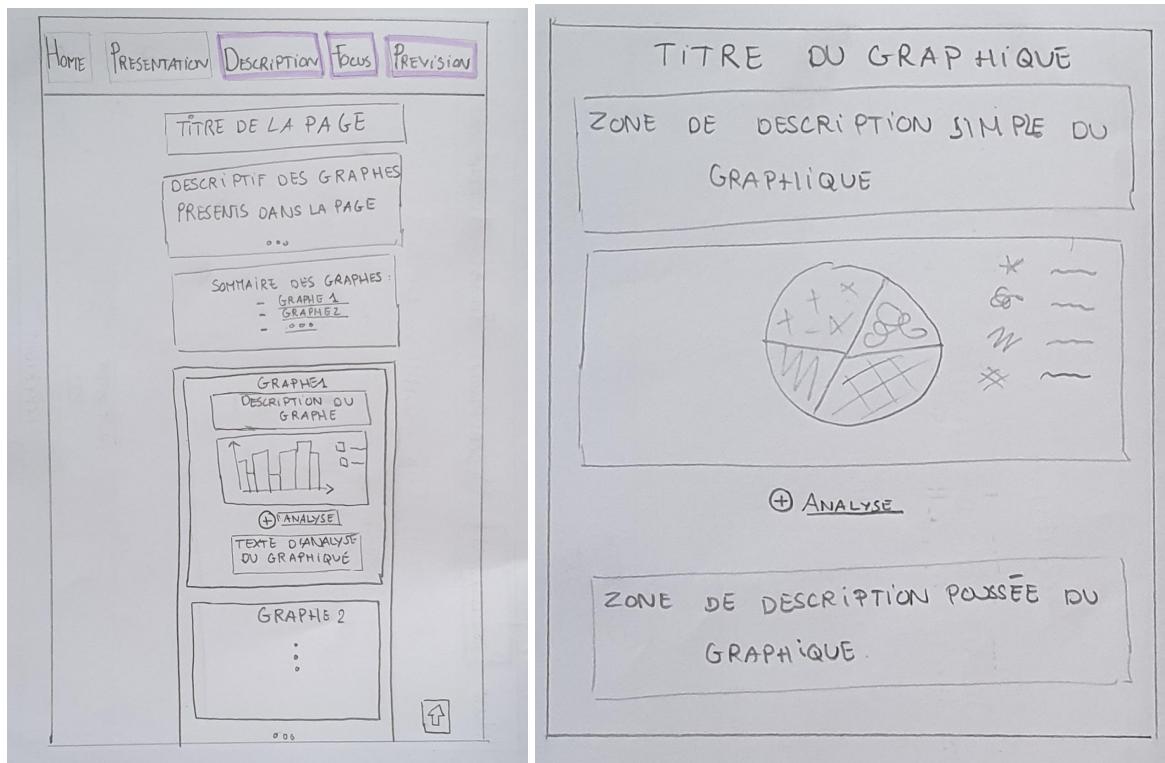
MAQUETTAGE

Voici le maquettage papier de notre site pour la disposition de nos graphiques et leur description.



La présentation des graphiques a été décomposée en 3 parties, qui sont :

1. Description
2. Effet covid
3. Prévision et simulation



L'interface web est constituée de trois pages qui résument notre analyse. La première page comporte la partie description qui présente les flux touristiques par monument et par site de 2017 à 2021 et le lien entre le site de Rome et Venise .Sur la deuxième page , on présente l'effet du covid sur le tourisme dans les 2 sites et l'évolution mensuelle du nombre de visites par monument et par site. La troisième page fait cas de la prévision.

PRÉSENTATION DES RÉSULTATS

Les graphiques et analyses de ceux-ci sont présentés dans l'interface disponible à l'adresse locale ci-dessous après avoir téléchargé le dossier “Site_web_covi_italie” sur votre machine, lancé le fichier “requirement.txt” qui permet le téléchargement de toutes les librairies python utiles pour le projet et exécuté le fichier “views.py”.

<http://127.0.0.1:5000/>

Nous avons aussi présenté des captures d'écrans en annexe.

Une description et une interprétation ont été rédigées pour chaque graphique.

CONCLUSION

Ce projet fut d'une grande richesse pour notre groupe. En effet, nous avons appris de nouvelles technologies et méthodes que nous n'avions pas vues en cours. A travers ce projet, nous avons appris également à travailler en équipe avec des idées divergentes. Nous avons cependant eu la chance de constituer nos groupes et de se mettre avec des personnes où une affinité était déjà présente. Cela nous a beaucoup aidé dans le fait que chacun puisse exprimer son avis à chaque décision prise par l'un des membres du groupe.

Afin d'aboutir à l'objectif que nous nous sommes fixés, nous sommes passés par plusieurs étapes à savoir la collecte des données, la préparation des données, l'analyse, la création des graphiques et pour finir la réalisation de l'interface web. Concernant la collecte des données, ce processus était tout nouveau pour tous les membres du groupe. Ainsi nous avons passé beaucoup de temps à nous informer et à nous former. Concernant la prévision et la simulation, nous avons pu mettre en œuvre des techniques apprises en cours dans la matière statistique.

Enfin, il ressort de cette étude menée sur les 2 sites touristiques italiens que le covid et toutes les restrictions autour du covid ont eu un fort impact sur le tourisme italien.

SITOGRAPHIE

Documentation librairie Selenium

<https://zestedesavoir.com/billets/2057/scraping-des-donnees-sur-une-page-web-en-python-avec-beautifulsoup-1/>

Documentation librairie Pandas

<https://pandas.pydata.org/docs/>

Documentation librairie plotly

<https://dash.plotly.com/>

Documentation librairie flask

<https://flask.palletsprojects.com/en/2.0.x/>

Site de documentation de PythonAnyWhere

<https://help.pythonanywhere.com/pages/>

Déploiement PythonAnywhere

https://www.youtube.com/watch?v=5jbdkOlf4cY&ab_channel=PrettyPrinted

Github

<https://docs.github.com/en>

Documentation git

<https://git-scm.com/doc>

Documentation prévision

<https://www150.statcan.gc.ca/n1/pub/12-539-x/2009001/seasonal-saisonnal-fra.htm>

<https://methodidacte.org/category/time-series/>

Documentation création site web

<https://www.pierre-giraud.com/>

Documentation Bootstrap

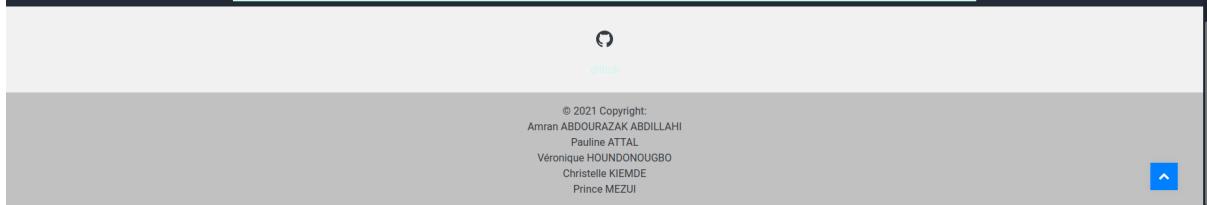
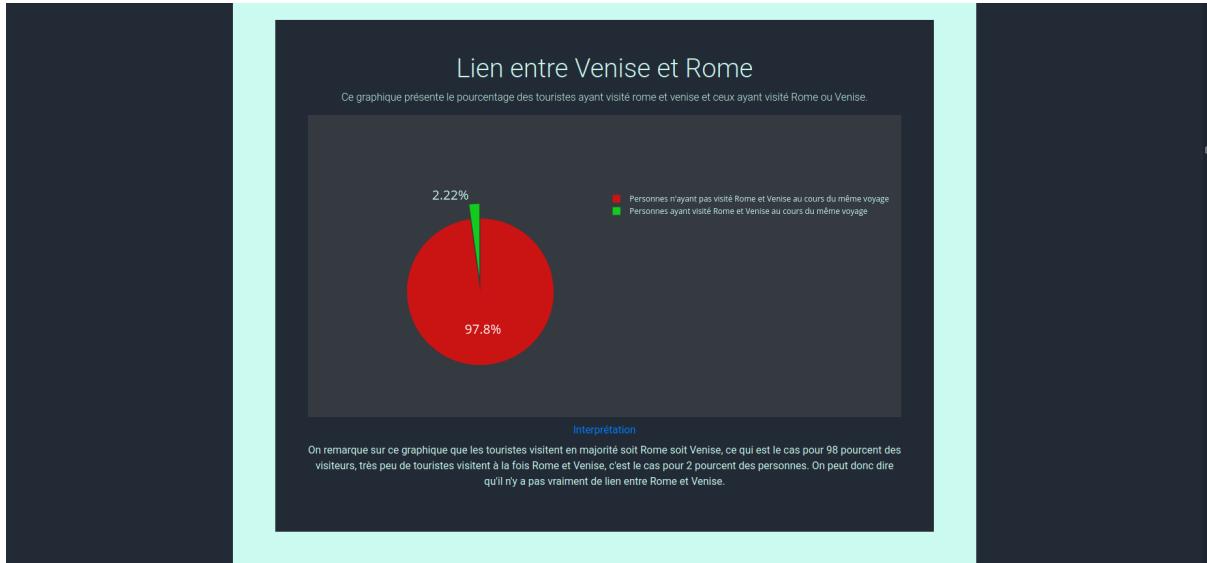
<https://getbootstrap.com/>

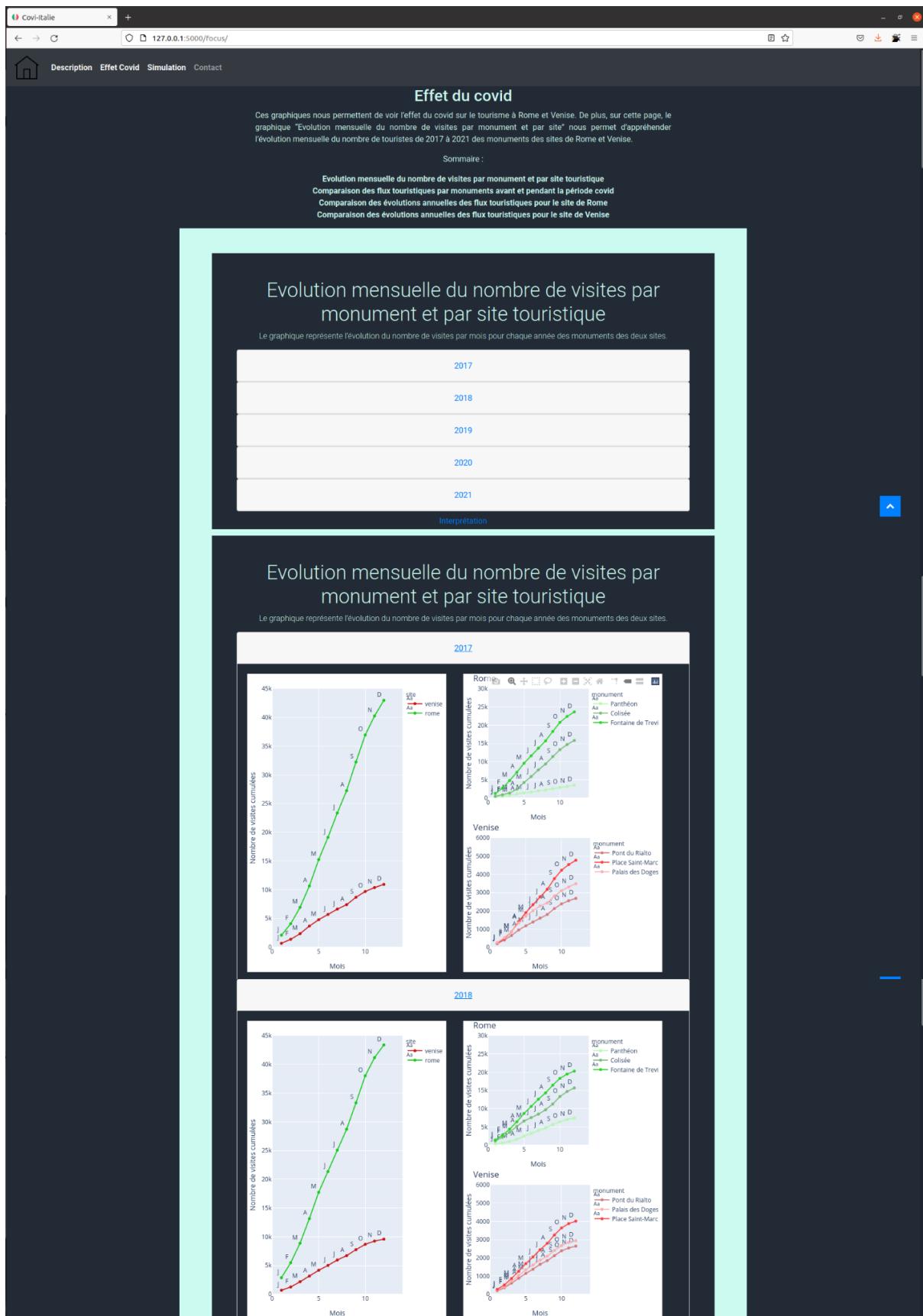
Forum

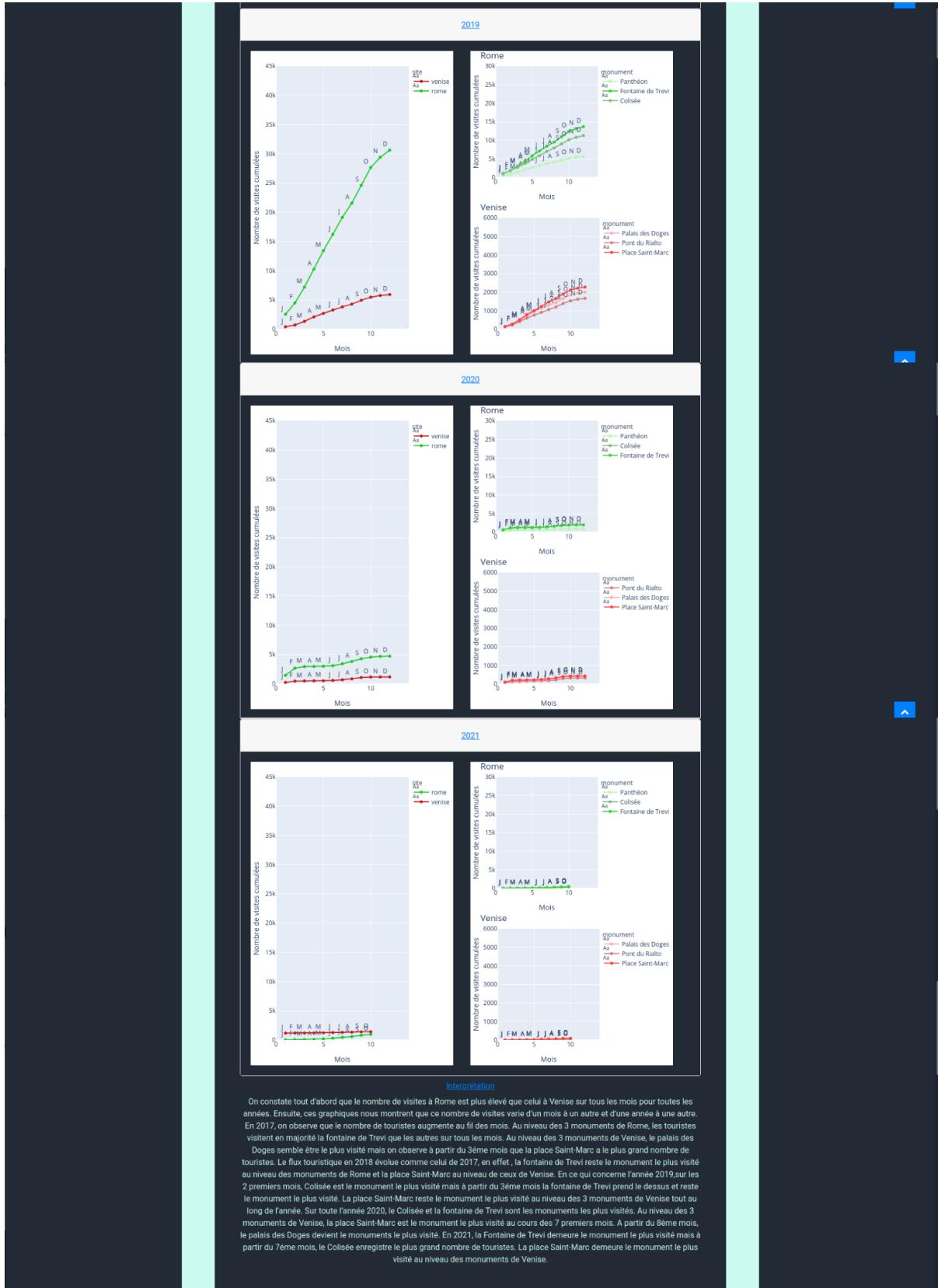
<https://stackoverflow.com/>

ANNEXES



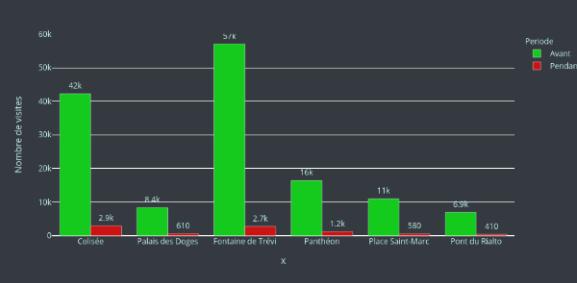






Comparaison des flux touristiques par monuments avant et pendant la période covid

Ce graphique présente le nombre de touristes ayant visité avant le covid et pendant le covid les monuments de Rome ou Venise.

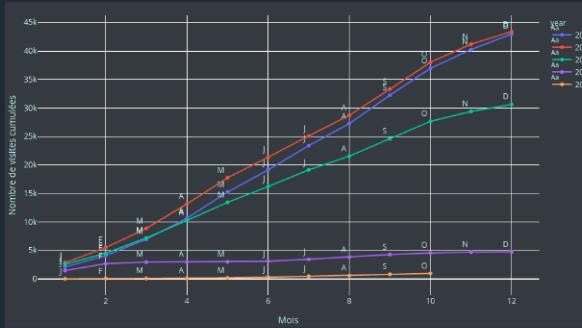


Interprétation

On constate tout d'abord qu'avant le covid le nombre de touristes est plus élevé que pendant le covid au niveau de tous les monuments. Ensuite, les chiffres nous montrent que pendant le covid, le Colisée et la fontaine de Trevi enregistrent le plus grand nombre de visites soient respectivement 2942 et 2749. Le graphique nous montre donc que le covid a un impact sur le tourisme à Rome et Venise.

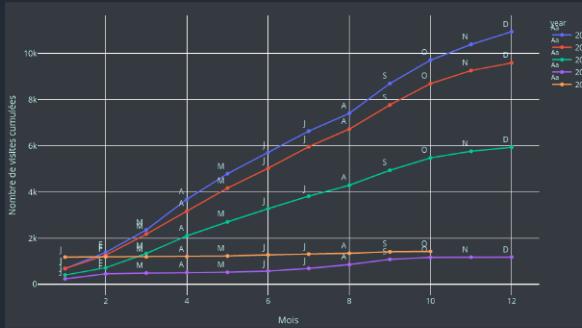
Comparaison des évolutions annuelles des flux touristiques pour le site de Rome

Ce graphique présente le nombre de touristes ayant visité le site de Rome de 2017 à 2021 (avant le covid et pendant le covid).



Comparaison des évolutions annuelles des flux touristiques pour le site de Venise

Ce graphique présente le nombre de touristes ayant visité le site de Venise de 2017 à 2021 (avant le covid et pendant le covid).



On remarque sur ce graphique que le nombre de touristes diminue au fur et à mesure des années. Les années 2017 et 2018 sont celles où Rome enregistre le plus grand nombre de touristes. Le nombre de touristes décroît faiblement jusqu'au 4ème trimestre de l'année 2019 avant de décroître fortement jusqu'à la fin de l'année. Pendant l'année 2020, très peu de touristes visitent Rome et en 2021, le nombre de touristes est très faible. L'année 2017 est celle où Venise enregistre le plus grand nombre de touristes.

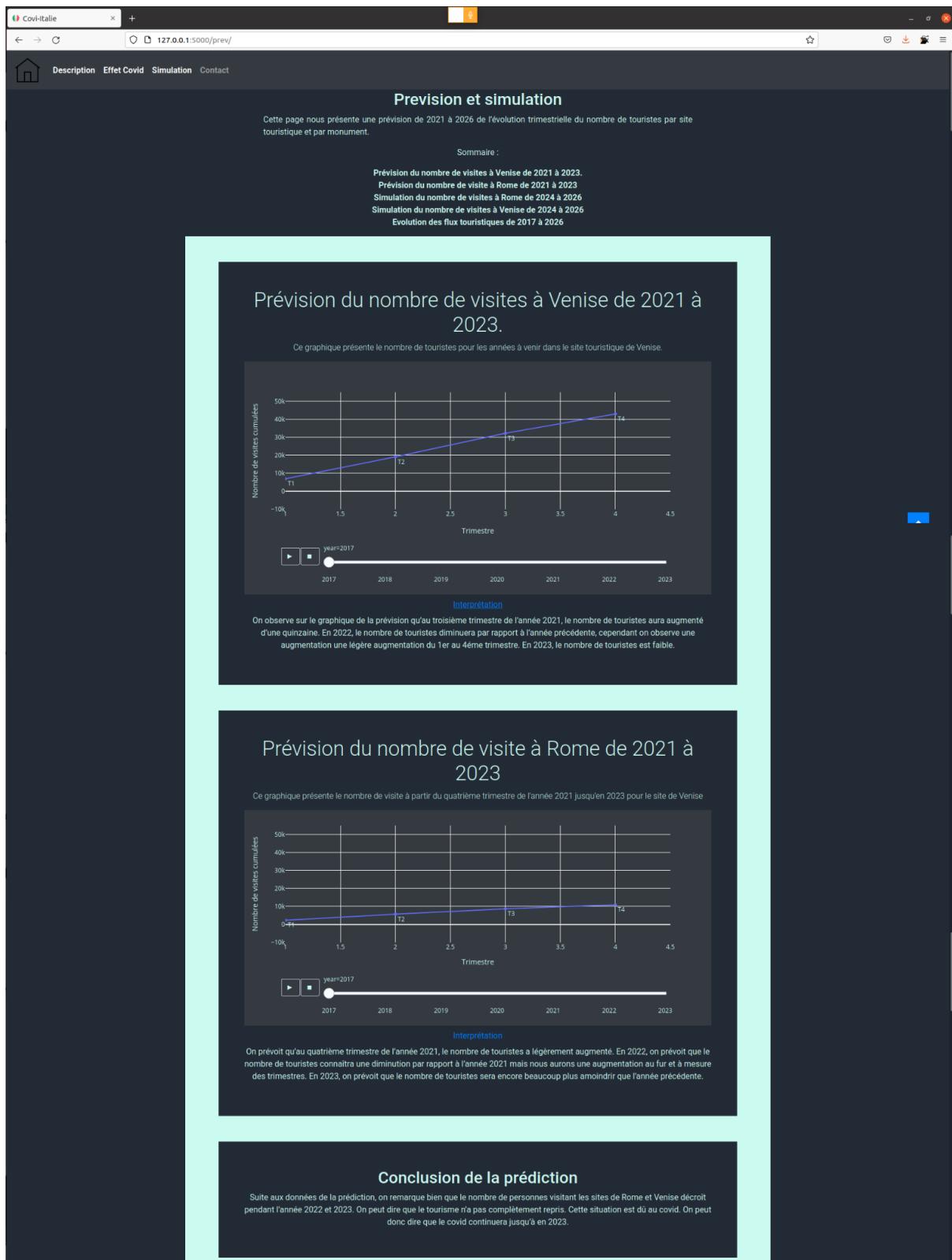
Le nombre de touristes diminue progressivement en 2018. En 2019, on observe une forte diminution du nombre de touristes.

Pendant l'année 2020, le nombre de touristes est très faible. Cependant, en 2021, on observe une légère augmentation du nombre de touristes. On peut donc conclure au vu de ces analyses que le covid a eu un impact sur le tourisme à Rome et à Venise.



QUESTION





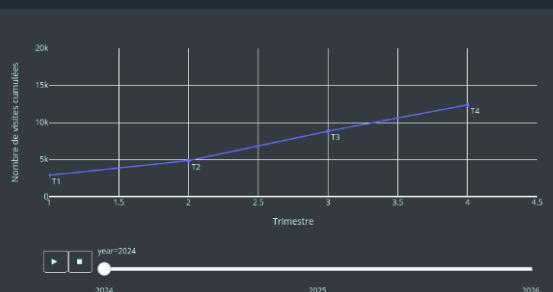
Simulation du nombre de visites à Rome de 2024 à 2026

Ce graphique présente le nombre de visite à partir du premier trimestre de l'année 2024 jusqu'en 2026 pour le site de Rome.



Simulation du nombre de visites à Venise de 2024 à 2026

Ce graphique présente le nombre de visite à partir du premier trimestre de l'année 2024 jusqu'en 2026 pour le site de Venise.



Interprétation

Le graphique de la simulation nous montre tout d'abord que le nombre de touristes à Rome et Venise augmentera de façon progressive de 2024 à 2026. Ensuite, on remarque que le nombre de visites à Rome sera toujours plus élevé que celui à Venise sur tous les trimestres pour toutes les années. De plus, on constate que le nombre de touristes augmente progressivement d'un trimestre à un autre pour chaque année.

Conclusion de la simulation

Suite aux données de la simulation, on remarque bien que le nombre de personnes visitant les sites de Rome et Venise croît de l'année 2024 à 2026. On peut dire que les activités ont repris, les gens peuvent voyager et donc le tourisme a repris. Cette reprise des activités est due au fait que la pandémie du covid a pris fin.

Evolution des flux touristiques de 2017 à 2026

Ce graphique présente les variations des flux touristiques des 2 sites de Rome et Venise tout au long nos années d'étude c'est à dire de 2017 à 2021 et présente les flux sur les années allant jusqu'en 2026.



Interprétation

Le graphique permet de regrouper les résultats de notre analyse et de voir encore une fois que la période Covid, de 2019 à 2023 a réellement été source de baisse des flux touristiques tant pour Rome que pour Venise. A partir de 2024, année supposée de retour à la normale, on remarque une nette hausse qui s'expliquerait par la réouverture totale des frontières et par l'envie que les gens auront de voyager à nouveau librement. Les années qui suivent le nombre de visites semble toujours augmenter. On pourrait expliquer cela par le fait qu'au fil du temps les quelques personnes qui étaient encore quelque peu réticentes ou fait de voyage immédiatement retrouvent peu à peu assurance ou encore que les personnes qui au sortir de la crise ne pouvaient pas en 2024 se payer ce voyage, ont pu ensuite regrouper assez d'argent.



gitHub

