

02/07/2025

FESTIVAL AVIGNON

CORIN Gaëtan
DOUSSIET Lou
VIDOR Fabien
BOUGEROLLES Jean
MARSILLE Antoine



PLAN

OBJECTIFS DU PROJET

ARCHITECTURE DU PROJET

ARCHITECTURE BDD

COMPARATIF DES SOLUTIONS

NOS CHOIX

CRITIQUES

AXES D'EVOLUTION

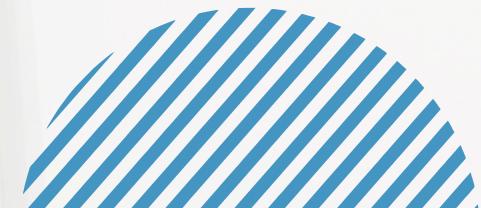
CE QUI A ETE FAIT

CONCLUSION

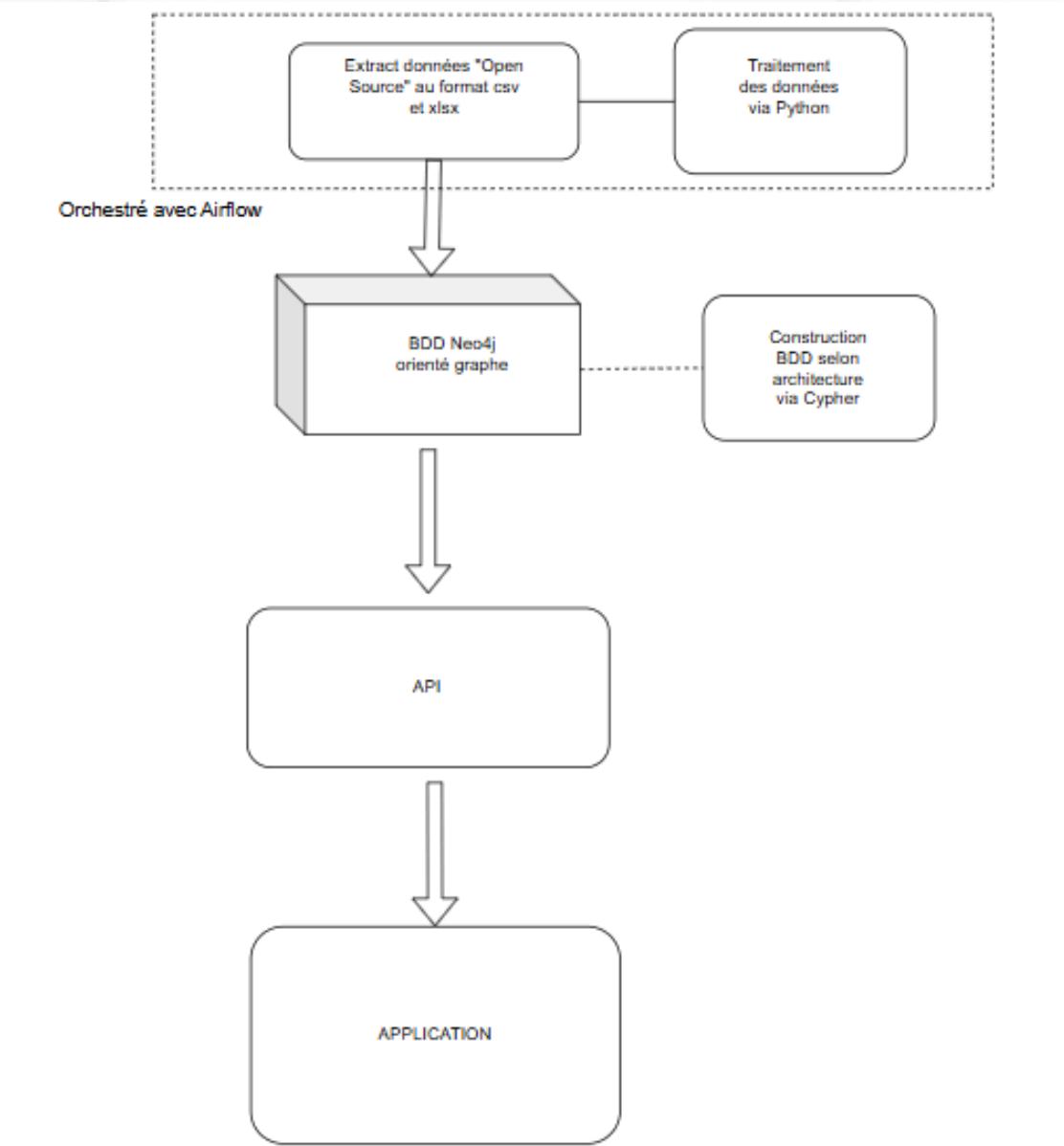


OBJECTIFS DU PROJET

- Optimiser l'expérience des visiteurs du Festival d'Avignon
- Réduire temps d'attente, optimiser les trajets et sensibiliser à l'éologie

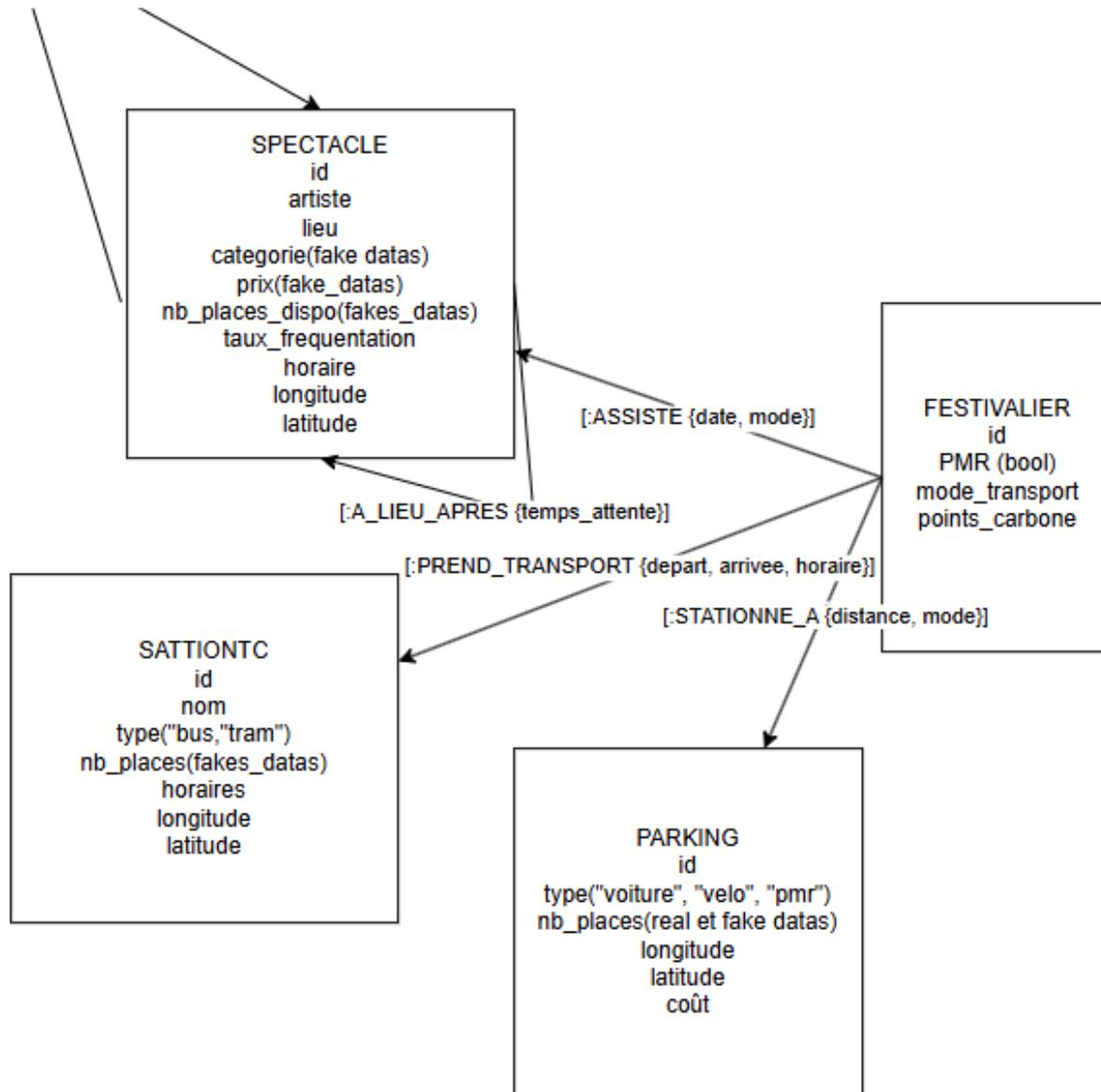


ARCHITECTURE DU PROJET



ARCHITECTURE BDD

[:PROCHE_DE {distance à pied, distance à vélo, distance en voiture}]



COMPARATIF DES SOLUTIONS

Base de données

MongoDB	Flexible, facile à utiliser	Moins performant pour données très relationnelles
Cassandra	Scalabilité, haute disponibilité	Pas de jointures, complexité requêtes
Neo4j	Excellente pour relations complexes, langage Cypher	Moins adapté aux grands volumes non relationnels
Hadoop HDFS	Stockage distribué massif	Accès par lots, pas en temps réel

Traitement des données

Apache Spark	Traitement en mémoire, rapide, modulaire	Consomme beaucoup de mémoire
Hadoop MapReduce	Très robuste, adapté aux gros volumes	Lent, lectures/écritures disque

COMPARATIF DES SOLUTIONS

Orchestration

Apache Airflow	Orchestration flexible avec DAGs, interface claire	Complexité de configuration initiale
Spark Structured Streaming	Traitement + orchestration streaming natif	Nécessite Spark, pas d'interface DAG

Application

React Native	Large communauté, facile avec JS	Moins fluide pour animations complexes
Kotlin (natif)	Performant Android, accès complet API	Code non réutilisable multiplateforme
Swift (natif)	Performant iOS, très intégré	Code non réutilisable multiplateforme

OUTILS SÉLECTIONNÉS POUR LE POC

PYTHON

- Simplicité
- Documentation
- Tests

NEO4J

- Adapté aux données
- Cohérence
- Tolérances aux partitions

KOTLIN

- Android
- Coroutines de programmation asynchrone
- Flexibilité

OUTILS SÉLECTIONNÉS POUR LA MISE EN PRODUCTION

SPARK

HADOOP
HDFS

NEO4J

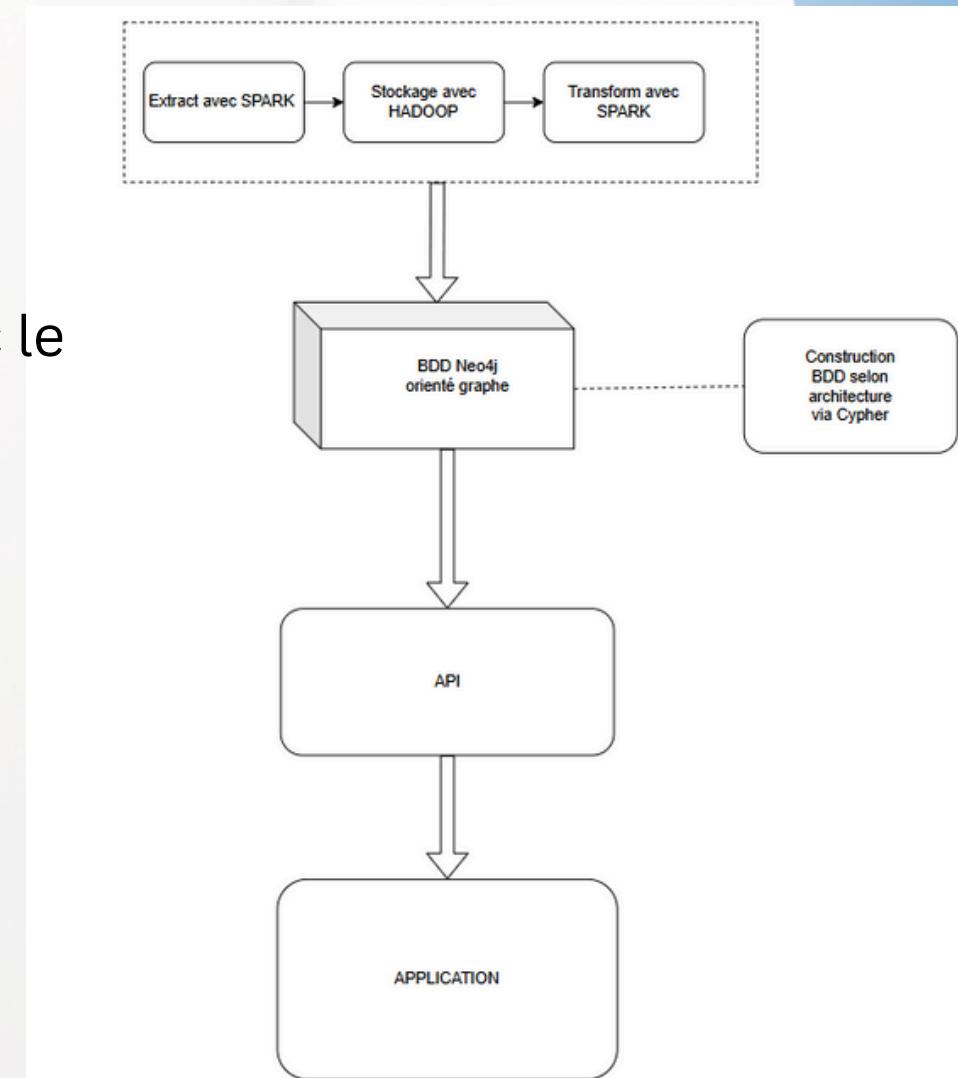
KOTLIN

CRITIQUES

- DONNEES SOURCES DU FESTIVAL PEUVENT ÊTRE AMÉLIORÉES
- PAS DE SYSTEME DE SUPERVISION
- PAS DE TESTS UNITAIRE SUR LES TRANSFORMATIONS SPARK

AXES + EVOLUTIONS

- Disponibilité et fiabilité des données => Partenariat avec le Festival
- Système de supervision
-



CE QUI A ETE FAIT

RÉCUPERATION DONNEES PARKINGS, STATIONNEMENT VÉLOS ET STATIONNEMENTS PMR

```
Entrée [5]: import requests
import pandas as pd

# RECUPERER LES PARKINGS
url = "https://www.datasud.fr/fr/datapusher/ws/default/usergroup506.eaae7b11-d531-47c4-be6e-75b0b9b9235e/all.xlsx?maxfeatures=-1&
response = requests.get(url)
with open("avignon-equipements-parkings.xlsx", 'wb') as f:
    f.write(response.content)
df = pd.read_excel("avignon-equipements-parkings.xlsx")
print(df.head())

# RECUPERER LES STATIONNEMENTS VELOS
url = "https://www.datasud.fr/fr/datapusher/ws/default/usergroup506.911f9ba7-442f-4bfd-b720-bcb9b1b4bd3d/all.xlsx?maxfeatures=-1&
response = requests.get(url)
with open("avignon-stationnement-pour-les-velos-arceaux.xlsx", 'wb') as f:
    f.write(response.content)
df = pd.read_excel("avignon-stationnement-pour-les-velos-arceaux.xlsx")
print(df.head())

# RECUPERER LES STATIONNEMENTS PMR
#Necessite un bot de scrapping qui va cliquer sur le bouton de download sur la carte
bot_result = "https://umap.openstreetmap.fr/fr/map/avignon-places-de-stationnement-pmr_19410#12/43.9467/4.8338"
# response = requests.get(url)
# with open("avignon_places_de_stationnement_pmr.csv", 'wb') as f:
#     f.write(response.content)
```

CE QUI A ETE FAIT

RÉCUPERATION DONNEES DE SPECTALES

```
import pdfplumber
import pandas as pd
import requests
from io import BytesIO

url = "https://festival-avignon.com/fr/telechargements/programme_FDA_79"
response = requests.get(url)
response.raise_for_status() # Pour lever une erreur si la requête échoue

# Charger le PDF en mémoire avec pdfplumber
with pdfplumber.open(BytesIO(response.content)) as pdf:
    page = pdf.pages[49] # page 50 (index 0-based)
    # Extraction de tous les tableaux de la page
    tables = page.extract_tables()

    table = tables[0]
    df = pd.DataFrame(table[1:], columns=table[0])
    df = df.applymap(lambda x: x.replace('\n', '') if isinstance(x, str) else x)
    df.to_csv("spectacles.csv", index=False, encoding="utf-8")
    print(df.head(5))
```

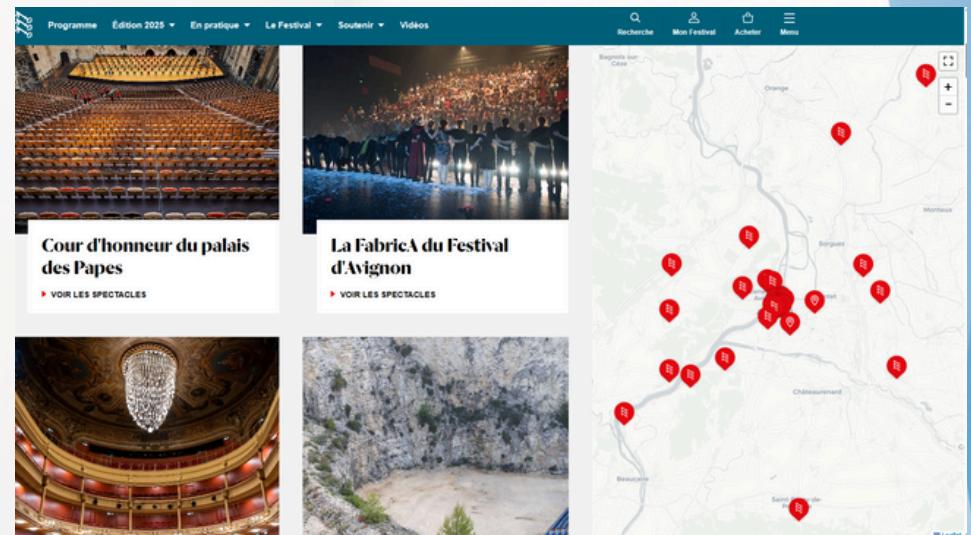
CE QUI A ETE FAIT

RÉCUPERATION DONNEES DES LIEUX

```
import pandas as pd
import requests
from bs4 import BeautifulSoup
from bs4 import BeautifulSoup as BS
import html
import json

url_lieu = "https://festival-avignon.com/fr/edition-2025/programmation/par-lieu"
response = requests.get(url_lieu)
soup = BeautifulSoup(response.text, "html.parser")
tag_places = soup.find("program-by-place")
raw_places = tag_places.get(":places")
decoded_places = html.unescape(raw_places)
places = json.loads(decoded_places)

data = []
for place in places:
    print({
        "id": place["id"],
        "title": place["title"],
        "slug": place["titleSlug"],
        "address": place["location"]["address"],
        "latitude": place["location"]["latitude"],
        "longitude": place["location"]["longitude"],
    })
    data.append({
        "id": place["id"],
        "title": place["title"],
        "slug": place["titleSlug"],
        "address": place["location"]["address"],
        "latitude": place["location"]["latitude"],
        "longitude": place["location"]["longitude"],
    })
df_places = pd.DataFrame(data)
print(df_places)
df_places.to_csv("lieu_lat_long.csv", index=False, encoding="utf-8")
```



CE QUI A ETE FAIT

MERGE CSV DONNEES SPECTACLES ET LIEUX

```
: import pandas as pd

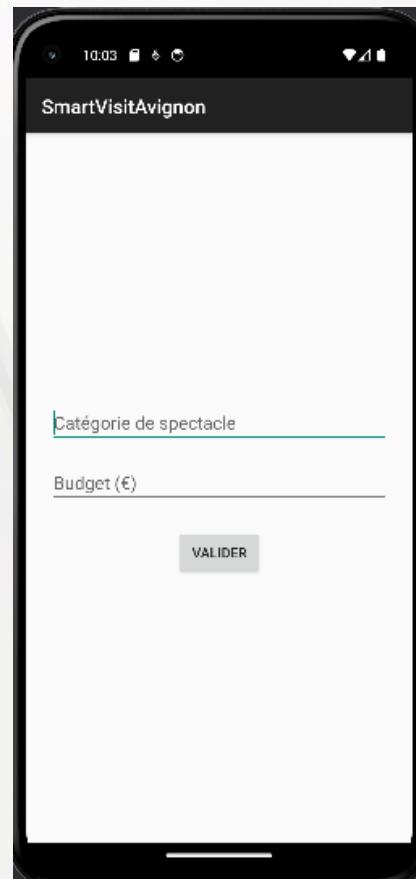
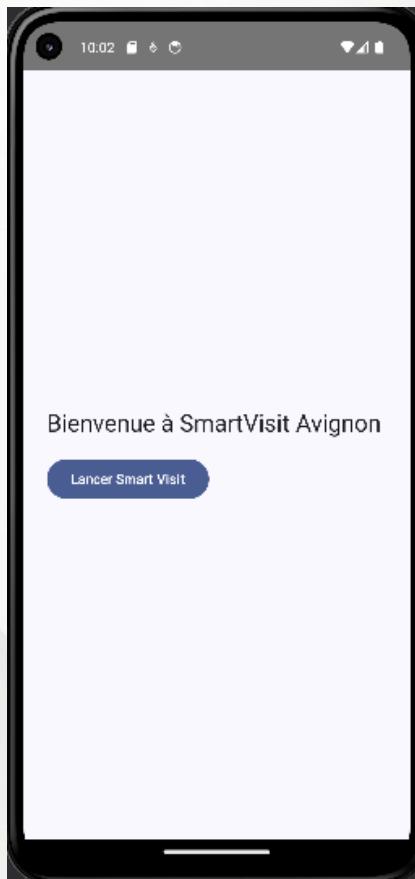
df_spectacle = pd.read_csv("cleaned_spectacles.csv", sep=";")
df_spectacle = df_spectacle.drop(columns=["PAGE", "Unnamed: 6"])
print(df_spectacle.head())

df_lieu = pd.read_csv("lieu_lat_long.csv")
df_lieu = df_lieu[["title", "address", "latitude", "longitude"]]
df_lieu = df_lieu.rename(columns={"title": "LIEUX"})
print(df_lieu.head())

df_merged = df_spectacle.merge(df_lieu, how="left", left_on="LIEUX", right_on="LIEUX")
print(df_merged.head())
df_merged.to_csv("merged_spectacles_and_lieu.csv", index=False, encoding="utf-8")
```

CE QUI A ETE FAIT

PAGES DE L'APPLICATION



CONCLUSION

MERCI !