OPENCLASSROOMS

SOUTENANCE

PROJET 9

PLAN

- Présentation du Projet
- Présentation du jeu de données
- Solutions proposées
- Résultats
- API et Applications
- Conclusion

PRÉSENTATION DU PROJET

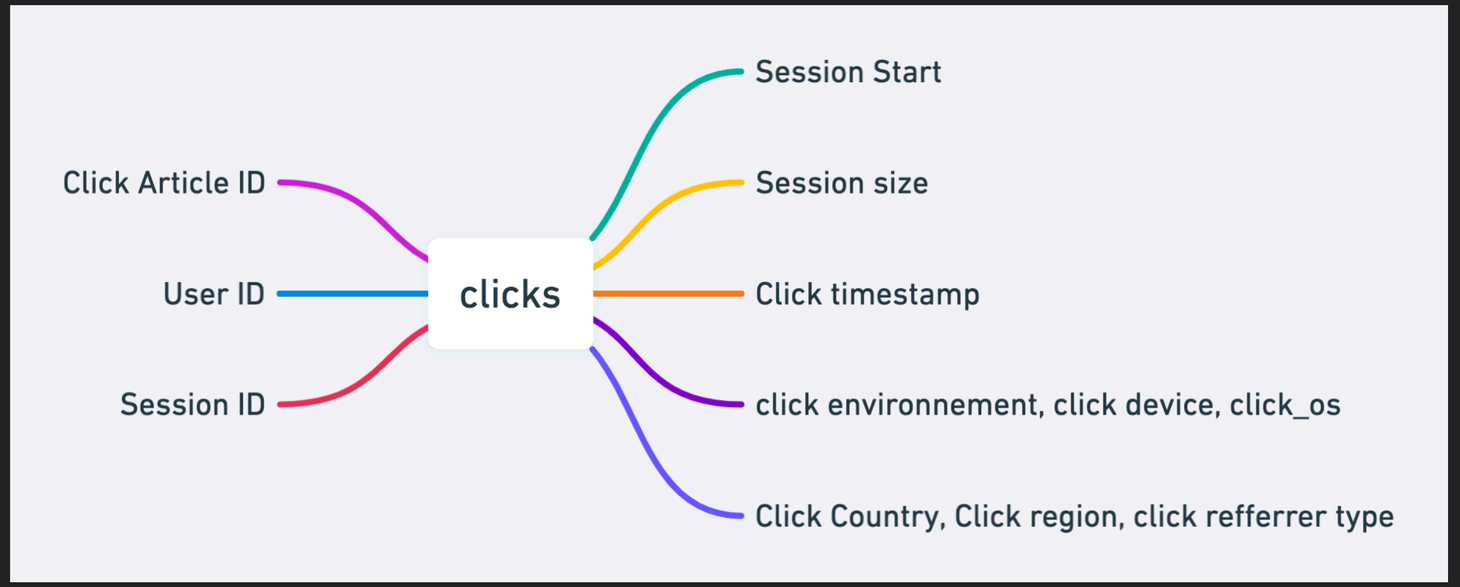
- CTO et cofondateur de la start-up « My Content ».
- Construction d’un premier MVP.
- Solution de recommandation d’articles et de livres à des particulier.
- Utiliser des données open-source.
- Permettre à l’utilisateur de recevoir une sélection de cinq articles.

MISSION

- Développer une première version de notre application mobile avec le système de recommandation.
- Stocker les scripts dans un dossier GitHub.
- Intégrer le système de recommandation à l'application mobile développée par Julien.

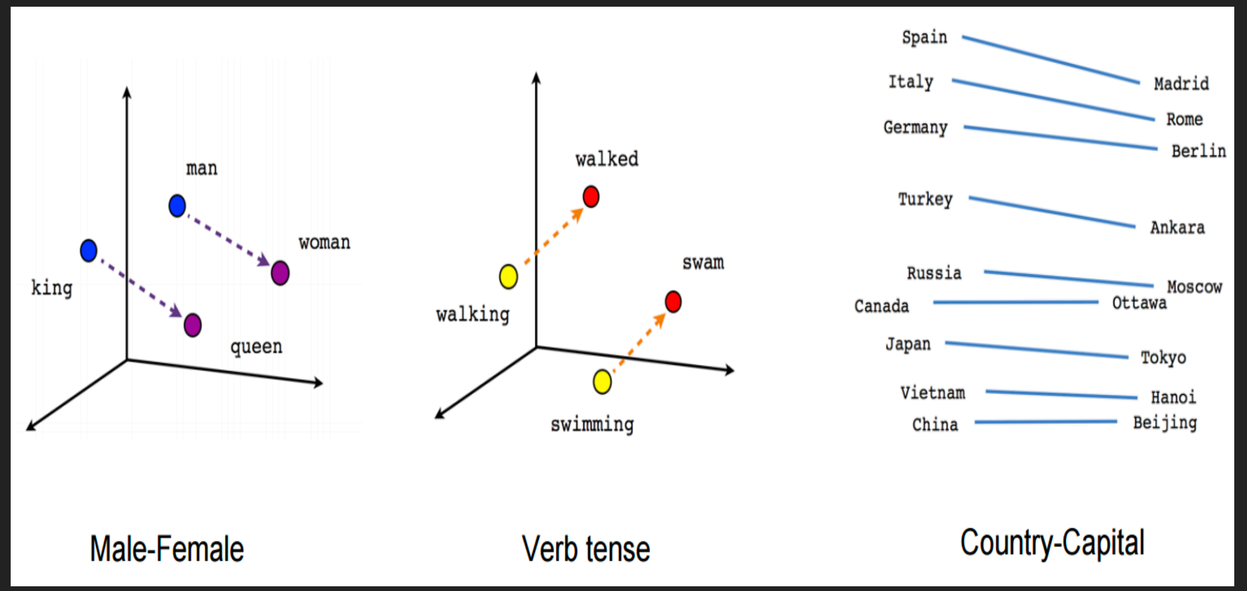
JEU DE DONNÉES

- Open-Source
- Contiennent des informations sur les articles.
- Temps de sessions, sur quels articles l'utilisateur a cliqué, etc …



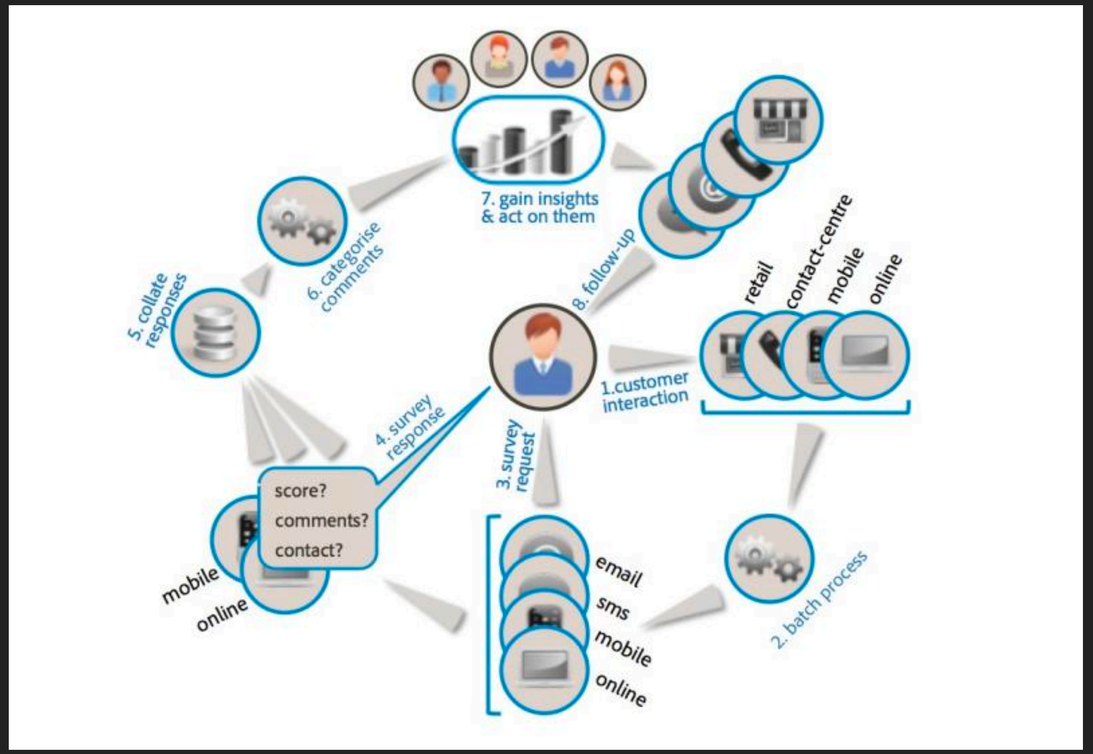
ARTICLE EMBEDDING

- Nous avons également un autre document « pickle » nommé Article_embeddings
- Technique en machine learning qui permet de représenter les mots ou les phrases d'un texte par des vecteur dans un espace vectoriel.

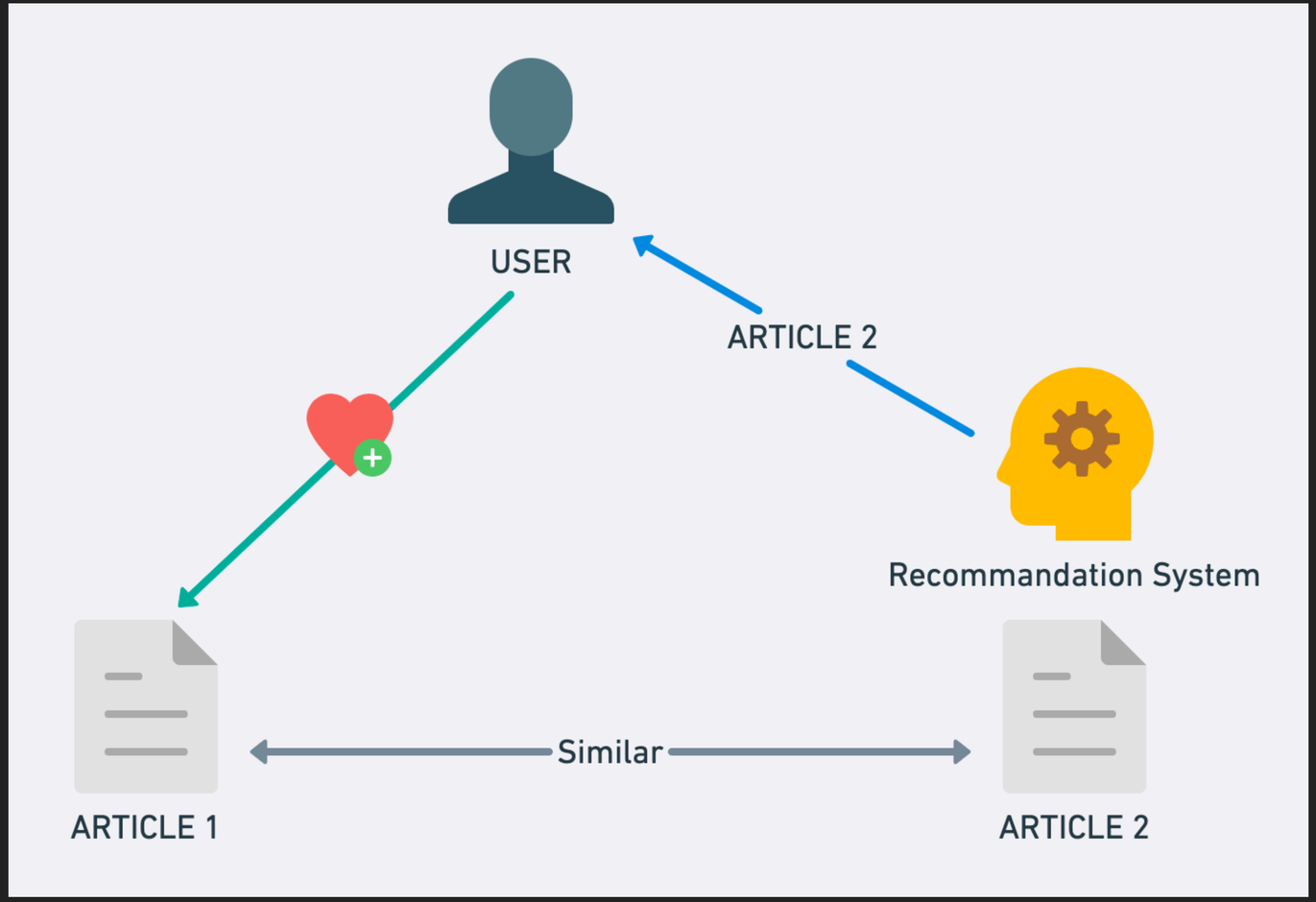


SOLUTIONS PROPOSÉES

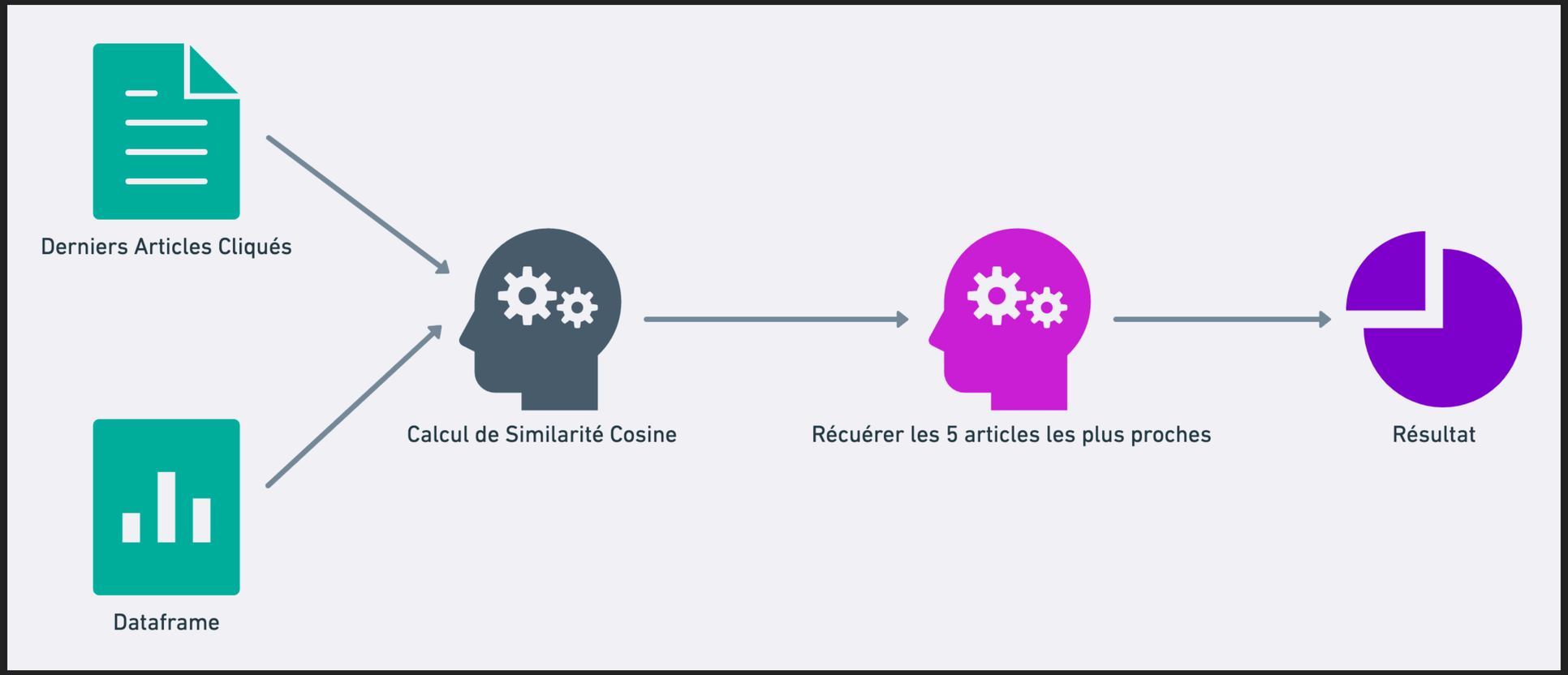
- Il existe plusieurs méthodes pour développer un système de recommandation nous allons en tester deux :
  - Content Based Filtering
  - Collaborative Filtering ou Filtrage Collaboratif



CONTENT BASED FILTERING



PRINCIPE TECHNIQUE



RÉSULTATS

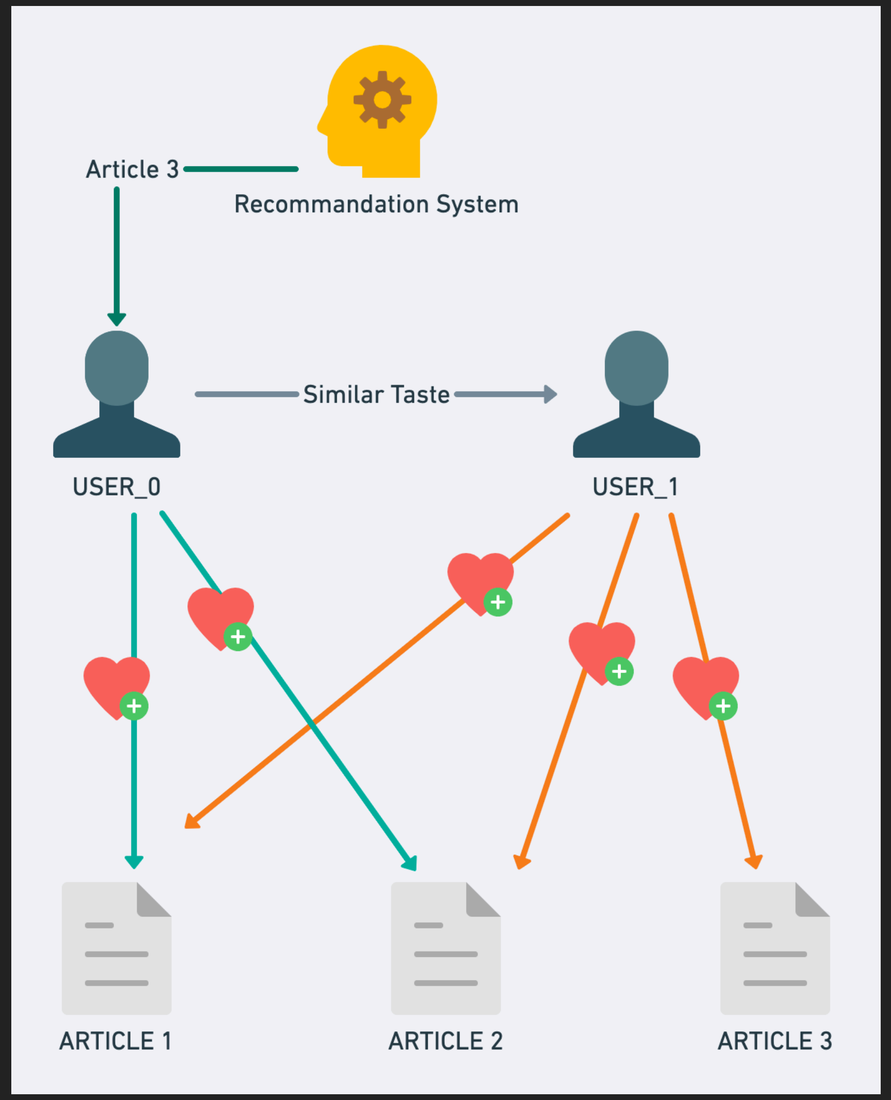
```
recommendFromArticle(article_id, 10)
```

```
[157514,
 159283,
 157554,
 162368,
 156355,
 161753,
 161447,
 156516,
 158173,
 162855]
```

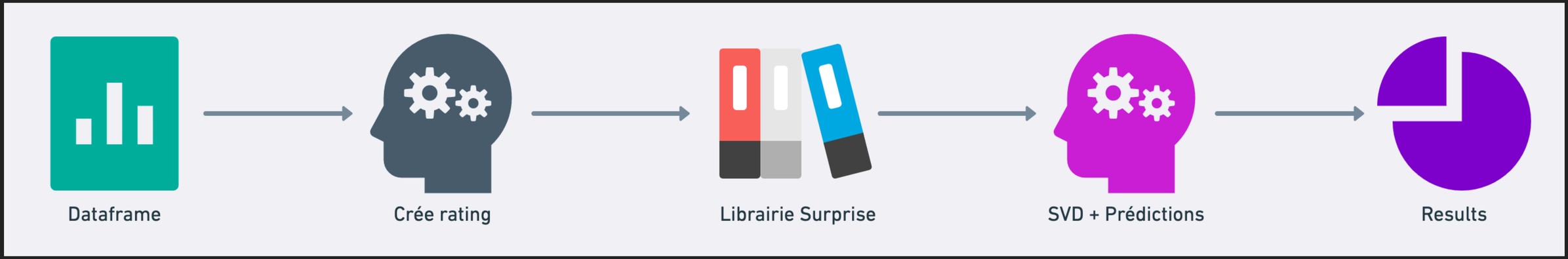
```
reco_base(user_id)
```

```
[30917, 168838, 225055, 66452, 275139]
```

COLLABORATIVE FILTERING



PRINCIPE TECHNIQUE



RÉSULTATS

```
findRecom(top_n, user_id)
```

```
[297, 125]
```

API FLASK

- Flask est un micro framework open-source de développement web en python.
- On créé notre API afin de pouvoir l'utiliser dans l'application de Julien.

```python
@app.route('/recommandation', methods=["POST"])
def reco():
    if request.method=='POST' :
        user = request.get_json(silent=True)
        id=user['userId']
        #user=json.dumps(user)
        #print(type(user))

        #id=user_id["id"]
        recommandation= reco_base(id)
        recom=json.dumps(recommandation)
        #listToStr = ' '.join([str(elem) for elem in recommandation])
        return recom
```

```python
import requests
import keras
import json
```

```python
url ='http://127.0.0.1:5000/recommandation'
```

```python
user=0
```

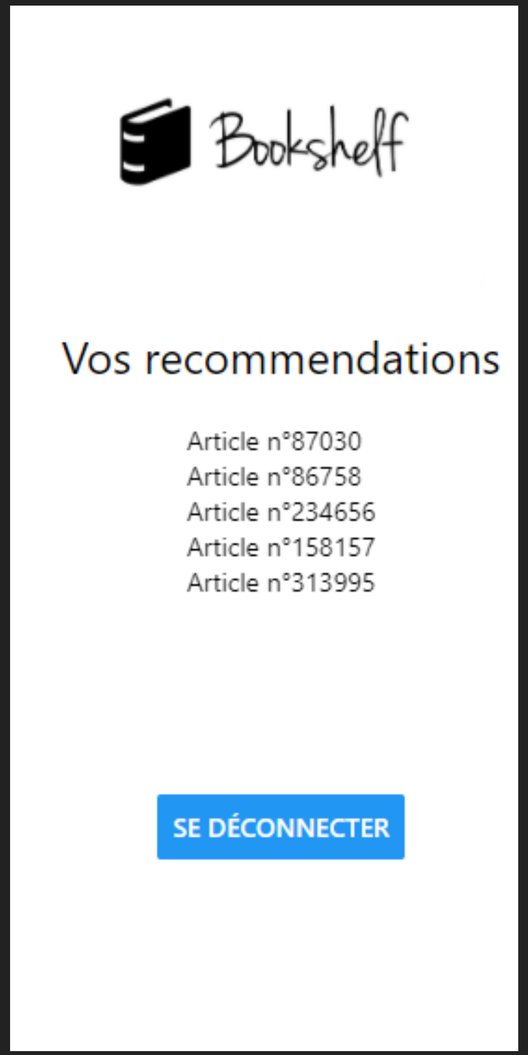
```python
r = requests.post(url, json={"id":0})
```

```python
r.text
```

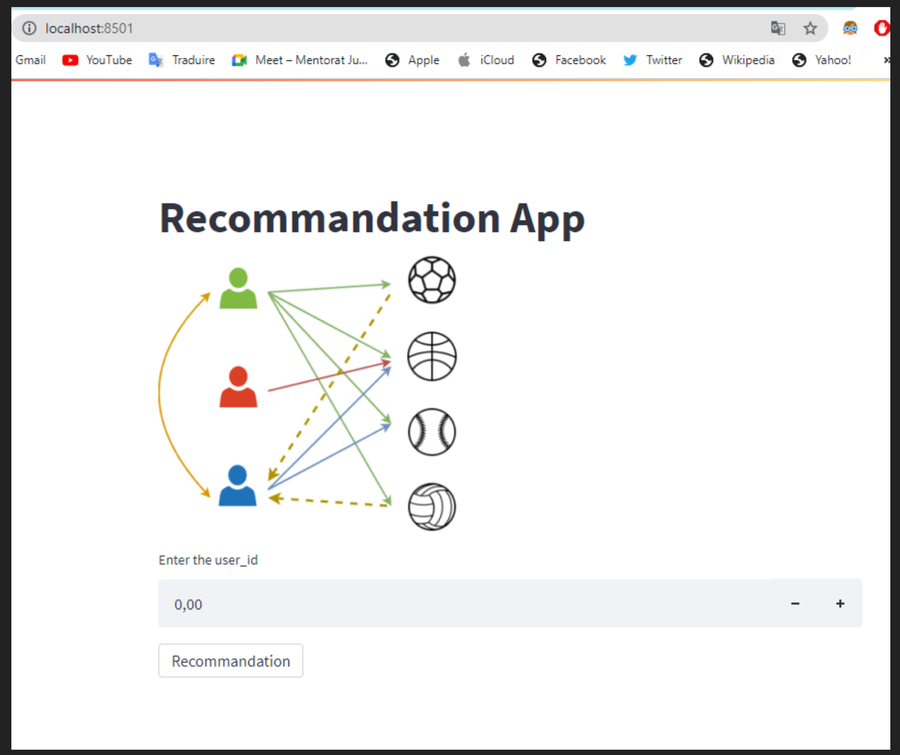
```
'86426 86758 236178 167735 310637'
```

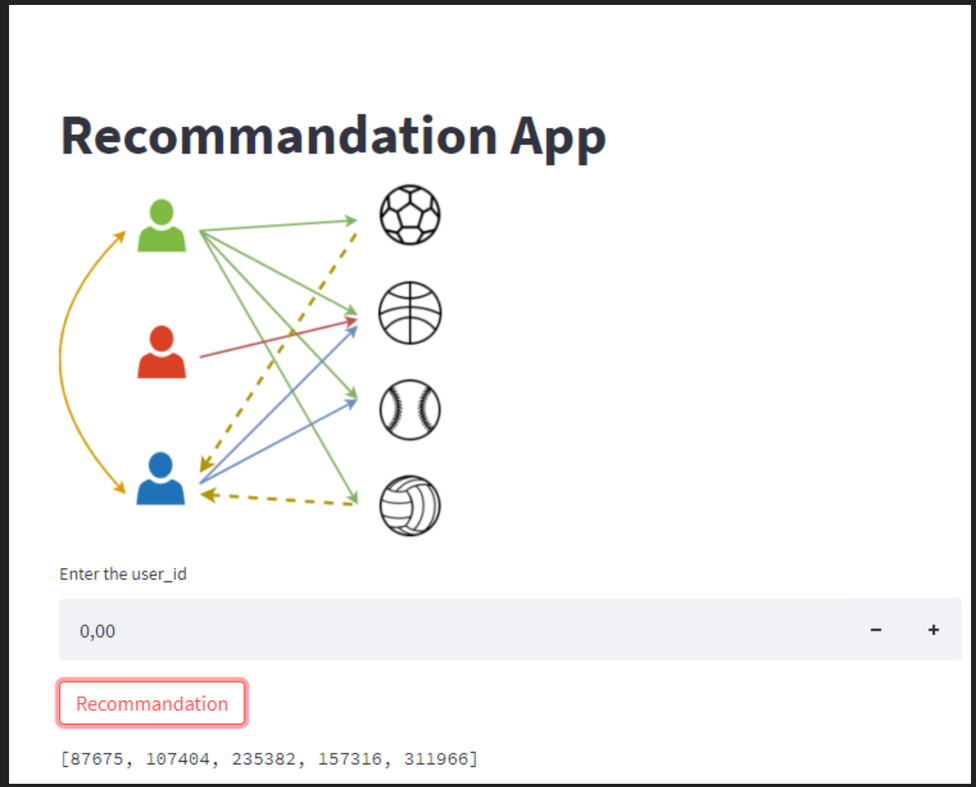
BOOKSHELF

- On a téléchargé l'application de Julien dans notre répertoire et lancé npm install dans le terminal.
- On intègre notre api dans l'application assez facilement et on peut lancer notre application avec npm start.
- Il a fallu modifier un morceau de code pour faire fonctionner l'application.



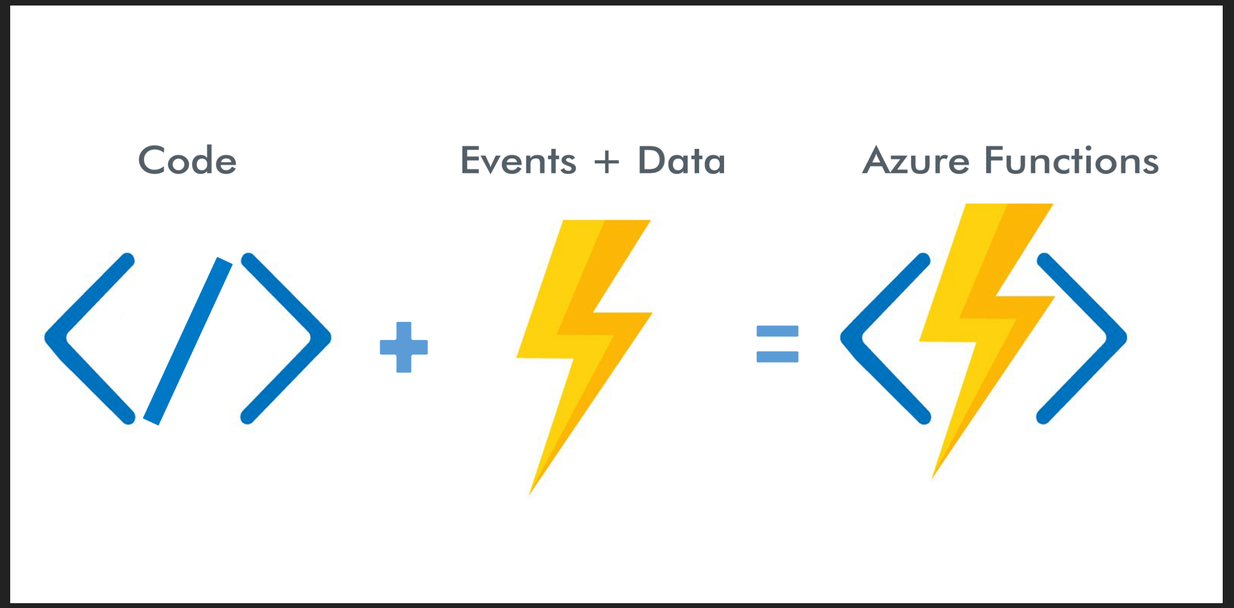
STREAMLIT


localhost:8501
Gmail
YouTube
Traduire
Meet – Mentorat Ju...
Apple
iCloud
Facebook
Twitter
Wikipedia
Yahoo!
Recommandation App
Enter the user_id
0,00
Recommandation


Recommandation App
Enter the user_id
0,00
Recommandation
[87675, 107404, 235382, 157316, 311966]

QU'EST- CE QUE AZURE FONCTION ?

- Plateforme d'application sans serveur.
- Permet aux développeurs d'héberger une application exécutable infrastructure.
- Facturé uniquement pour les ressources utilisées.
- S'exécute uniquement en réponse à un Trigger.



AZURE FONCTION

- Création d’un projet local sur visual studio avec azure fonction.
- Traitement du système de recommandation dans init.py.
- Test en local
- Déploiement Azure fonction.

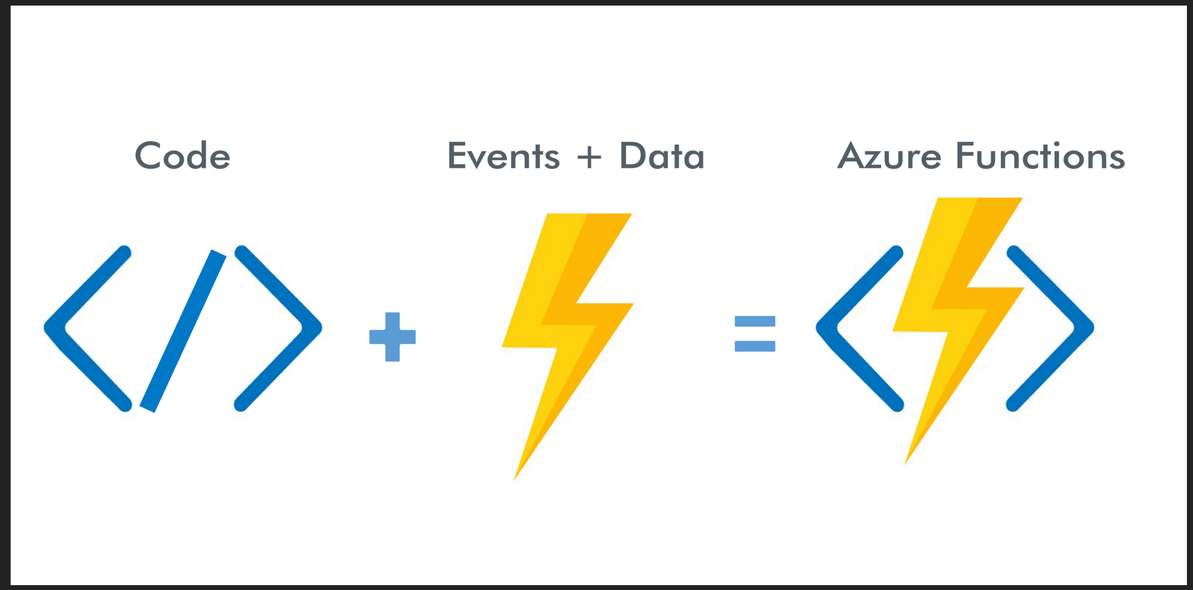


SCHÉMA D'ARCHITECTURE

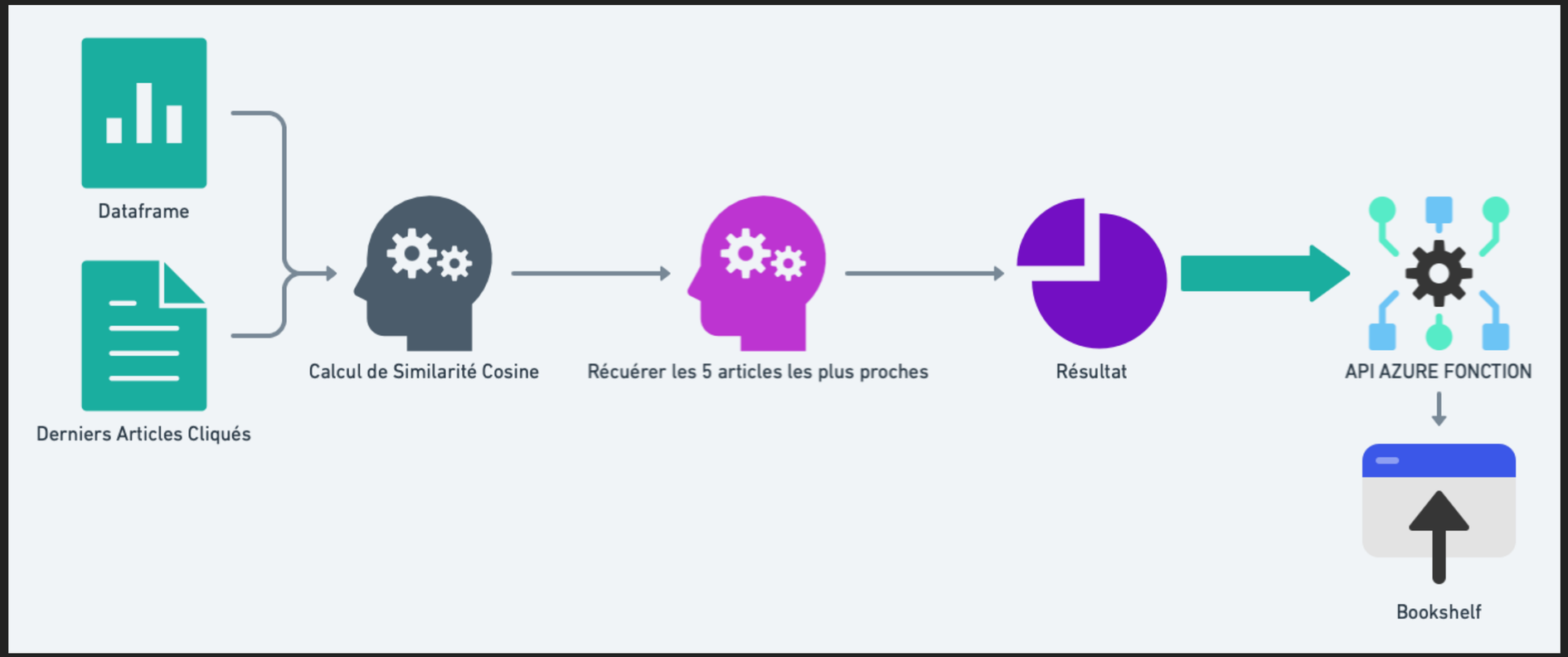
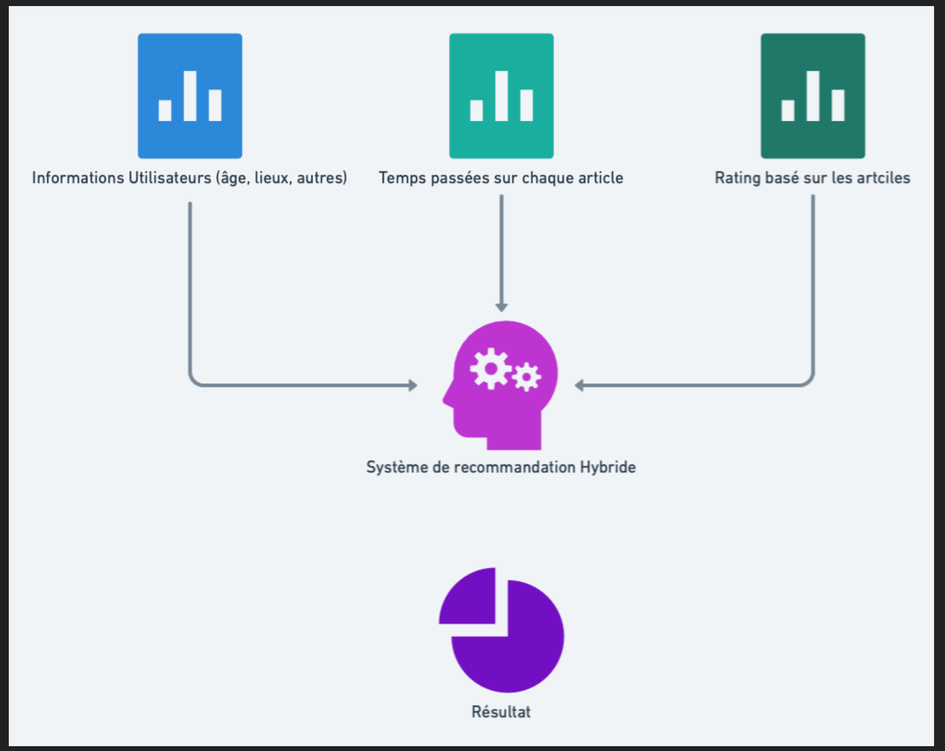


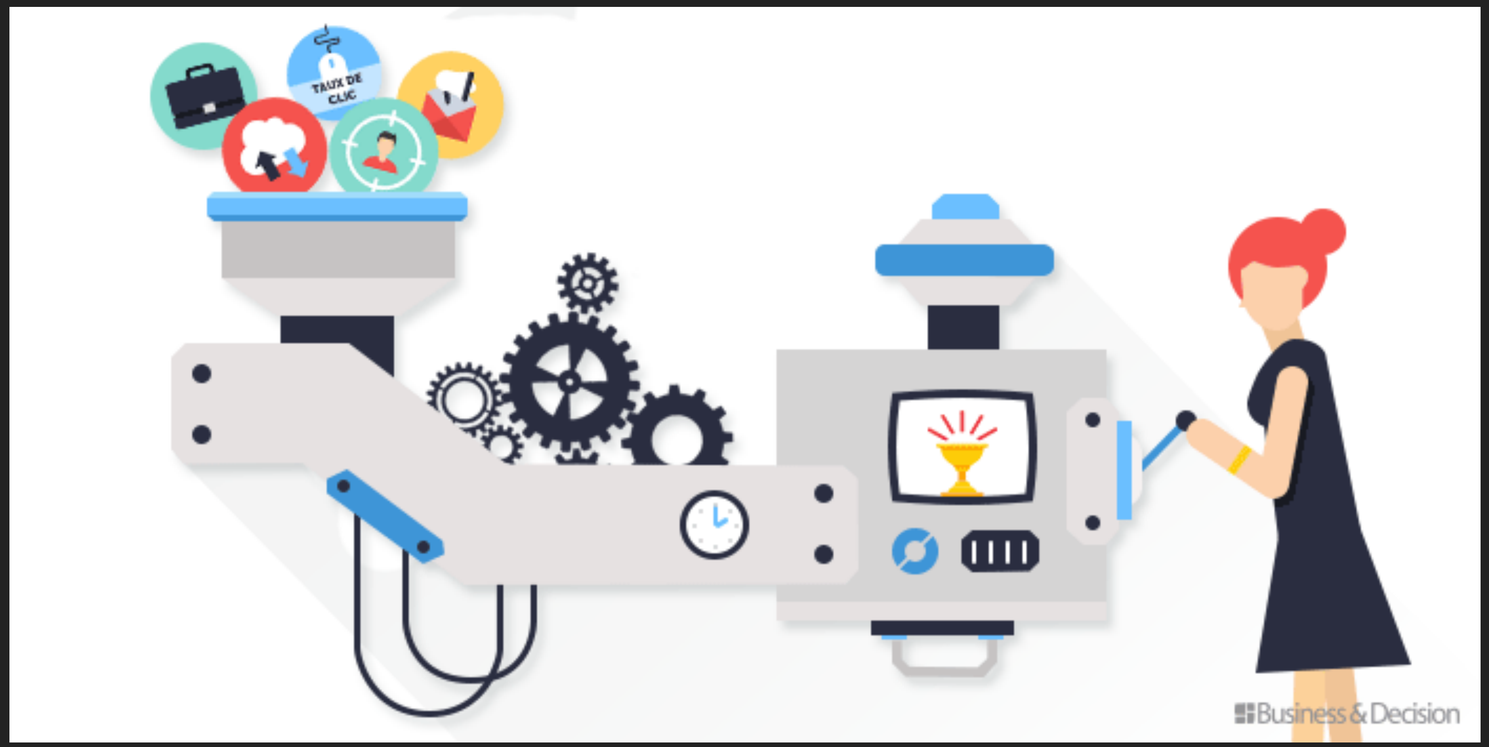
SCHÉMA D'ARCHITECTURE FUTURE

- Nous pourrions proposer une architecture hybride. Se basant à la fois sur le contenu et à la fois sur les utilisateurs similaires.
- Il faut pouvoir faire tourner le moteur de rechercher périodiquement ou à chaque fois que la base de données est complétée.



CONCLUSION

- Nous avons su répondre aux besoins concernant le moteur de recommandation.
- Nous avons testé deux différents types de moteur de recommandation.
- Stockés les scripts dans un dossier GitHub.
- Intégrer le système de recommandation à l’application.



MERCI POUR VOTRE ATTENTION