Partie 1 : Recherche et analyse des technologies

1. Présentation des technologies Cloud

Amazon Web Services (AWS)

Définition : AWS est une plateforme Cloud offrant plus de 200 services, allant du stockage à l'analyse de données, en passant par le machine learning.

Stockage (Amazon S3) :

Définition : Service de stockage objet scalable pour les données non structurées.

Cas d'usage : Netflix utilise S3 pour stocker ses vidéos et métadonnées de visionnage.

Data Warehouse (Amazon Redshift) :

Définition : Entrepôt de données rapide et scalable pour les grandes analyses SQL.

Cas d'usage : Lyft analyse en temps réel les trajets et les tarifs dynamiques.

Traitement en temps réel (AWS Lambda) :

Définition : Service de calcul serverless exécutant des fonctions en réponse à des événements.

Cas d'usage : The Washington Post génère des pages dynamiques selon les comportements des lecteurs.

Microsoft Azure

Définition : Plateforme Cloud hybride de Microsoft, proposant des services d'analyse, IA, bases de données, etc.

Analyse (Azure Synapse Analytics) :

Définition : Solution intégrée combinant Data Warehouse et Big Data.

Cas d'usage : Starbucks optimise ses offres en fonction des commandes en temps réel.

Base NoSQL (Azure Cosmos DB) :

Définition : Base de données NoSQL multi-modèle, distribuée à l'échelle mondiale.

Cas d'usage : Slack utilise Cosmos DB pour des performances rapides dans ses messages.

Google Cloud Platform (GCP)

Définition : Plateforme Cloud de Google axée sur l'intégration Big Data, IA, et services d'infrastructure.

Analyse Big Data (BigQuery) :

Définition : Data Warehouse serverless permettant l'analyse rapide de grands ensembles de données.

Cas d'usage : Spotify personnalise des playlists en analysant les habitudes de ses utilisateurs.

Pipeline (Dataflow) :

Définition : Service de traitement de flux de données en temps réel ou par batch.

Cas d'usage : King traite les données des joueurs de Candy Crush pour surveiller les performances des jeux.

2. Étude des architectures de données modernes

Data Lakes

Définition : Réserves de données où les données brutes sont stockées dans leur format natif, prêtes pour un traitement ultérieur.

Exemple : Airbnb utilise un Data Lake (AWS S3) pour stocker des données clients et transactions, alimentant ses modèles de machine learning.

Data Warehouses

Définition : Bases de données optimisées pour organiser et analyser des données structurées.

Exemple : Uber utilise BigQuery pour surveiller les performances des conducteurs et ajuster les prix.

Pipelines ETL/ELT

ETL (Extract, Transform, Load) : Processus où les données sont extraites, transformées, puis chargées dans un entrepôt de données.

ELT (Extract, Load, Transform) : Variante où les données sont d'abord chargées dans le système, puis transformées selon les besoins analytiques.

Exemple : Netflix utilise des pipelines ETL pour analyser les comportements des utilisateurs et personnaliser les recommandations.

Voici une version enrichie de la **Partie 2 : Conception de l'architecture** avec des exemples de code et de résultats issus de votre projet :

Partie 2 : Conception de l'architecture

Contexte et objectifs

L'objectif est de concevoir un système robuste permettant de traiter et analyser en temps réel les données provenant de capteurs IoT, en détectant les anomalies critiques et en les visualisant de manière compréhensible.

Choix des services et outils

1. Collecte et ingestion des données

Les données issues des capteurs IoT sont fournies sous la forme d'un fichier CSV. Voici un exemple de chargement et d'exploration des données :

```
python
import pandas as pd

# Charger les données à partir du fichier CSV
data = pd.read_csv('/content/iot_telemetry_data.csv')

# Aperçu des premières lignes
print(data.head())
```

Extrait des données :

ts	device	co	humidity	light	lpg	motion	smoke	temp
1594512094.0	b8:27:eb:bf:9d:51	0.004956	51.0	False	0.007651	False	0.020411	22.7
1594512094.7	00:0f:00:70:91:0a	0.002840	76.0	False	0.005114	False	0.013275	19.7

Une fois les données vérifiées, elles sont envoyées à Google Pub/Sub pour simuler un flux de messages.

Exemple de publication dans Pub/Sub :

```
python
```

```
from google.cloud import pubsub_v1

project_id = 'ton-id-de-projet'
topic_id = 'ton-topic-id'

publisher = pubsub_v1.PublisherClient()
topic_path = publisher.topic_path(project_id, topic_id)

def publish_data_to_pubsub(data):
    data_bytes = str(data).encode('utf-8')
    publisher.publish(topic_path, data_bytes)
    print(f"Message publié : {data}")

# Exemple d'envoi de données
data_sample = {'timestamp': '2024-11-21T12:00:00', 'temperature': 22.5, 'humidity': 60}
publish_data_to_pubsub(data_sample)
```

Sortie console :

```
Message publié : {'timestamp': '2024-11-21T12:00:00', 'temperature': 22.5, 'humidity': 60}
```

2. Traitement des données et détection des anomalies

Une fois les données publiées, elles sont traitées pour détecter des anomalies (comme une température élevée).

Exemple de script pour détecter les anomalies :

```
  python
threshold_temp = 30  # Définir le seuil critique

for index, row in data.iterrows():
```

```
    if row['temp'] > threshold_temp:
        print(f"Anomalie détectée : Timestamp = {row['ts']}, Température = {row['temp']}")
```

Sortie console (exemple d'anomalies détectées) :

Anomalie détectée : Timestamp = 1594852498.371877, Température = 30.1

Anomalie détectée : Timestamp = 1594852510.269512, Température = 3

3. Visualisation des données

Les résultats des analyses sont visualisés sous forme de graphiques interactifs à l'aide de Matplotlib.

Exemple de visualisation avec un nuage de points :

python

```
import matplotlib.pyplot as plt

plt.scatter(data['ts'], data['temp'], c=['red' if t > threshold_temp else 'blue' for t in data['temp']])
plt.title("Visualisation des températures en fonction du temps")
plt.xlabel("Horodatage")
plt.ylabel("Température")
plt.show()
```

Résultat attendu :

- Points rouges : Températures critiques (au-dessus du seuil).
- Points bleus : Températures normales.

Étapes principales du flux de données :

1. Les données des capteurs IoT sont chargées et explorées.
2. Les données sont envoyées en flux continu vers Google Pub/Sub.
3. Les anomalies sont identifiées en temps réel (températures élevées).
4. Les anomalies et les tendances sont visualisées de manière interactive.