## **Shivani Sinha, IUBH**

### **Project Overview**

The aim of this project is to build a batch-processing-based data-intensive application using Google Cloud Platform (GCP) services. The application will ingest a large time-series dataset, perform batch processing, store the data securely, and deliver aggregated insights to a visualization dashboard. The application architecture emphasizes modularity, scalability, maintainability, and data governance.

### **Chosen Dataset**

* **Dataset Name**:Synthetic ride-sharing emissions dataset (self-generated)
* **Justification**:  
  + Includes 1000+ time-referenced records with fields like ride ID, vehicle type, distance, city, and timestamp.
  + Simulates real-world ride emissions with preassigned vehicle categories.
  + Fully customizable for reproducibility and integration into GCP-based batch pipelines.

### **Microservices and Responsibilities**

1. **Data Ingestion Microservice**
   * **Technology**: Cloud Scheduler → Pub/Sub → Workflows
   * **Task**: Scheduler triggers a Pub/Sub message weekly, which initiates a Workflow to load new data from Cloud Storage into BigQuery.
   * **Justification**: Replaces Cloud Functions with Workflows to remain fully within the GCP free tier. Pub/Sub decouples the trigger logic.
2. **Data Storage Microservice**
   * **Technology**: Google Cloud Storage (raw), BigQuery (processed).
   * **Task**: Store uploaded CSV (`rides.csv`) in GCS; load it into a BigQuery table (`ride\_emissions`) using Workflows.
   * **Justification**: BigQuery supports partitioning and SQL analytics on large datasets. GCS provides cost-effective staging.
3. **Data Preprocessing and Aggregation Microservice**
   * **Technology**: BigQuery SQL View
   * **Task**: Calculate `co2\_grams` using emission factors for each vehicle type. This is done through a scheduled or on-demand BigQuery view.
   * **Justification**: Avoids Python/Dataproc. SQL-based transformation is more cost-efficient and manageable within BigQuery.
4. **Data Delivery Microservice**
   * **Technology**: Looker Studio connected to BigQuery.
   * **Task**: Deliver visual insights — such as total emissions, emissions by city and vehicle type, and monthly trends — using interactive charts.
   * **Justification**: Looker Studio is free, and native integration with BigQuery is seamless.

### **Reliability, Scalability, and Maintainability**

* **Reliability**: Workflow retries can be configured; BigQuery ensures durable storage.
* **Scalability**: GCP serverless services (Workflows, Pub/Sub, BigQuery) scale automatically with load.
* **Maintainability**: Each service is modular and configured via UI or YAML for reproducibility. GitHub can be used to version the Workflow definition.

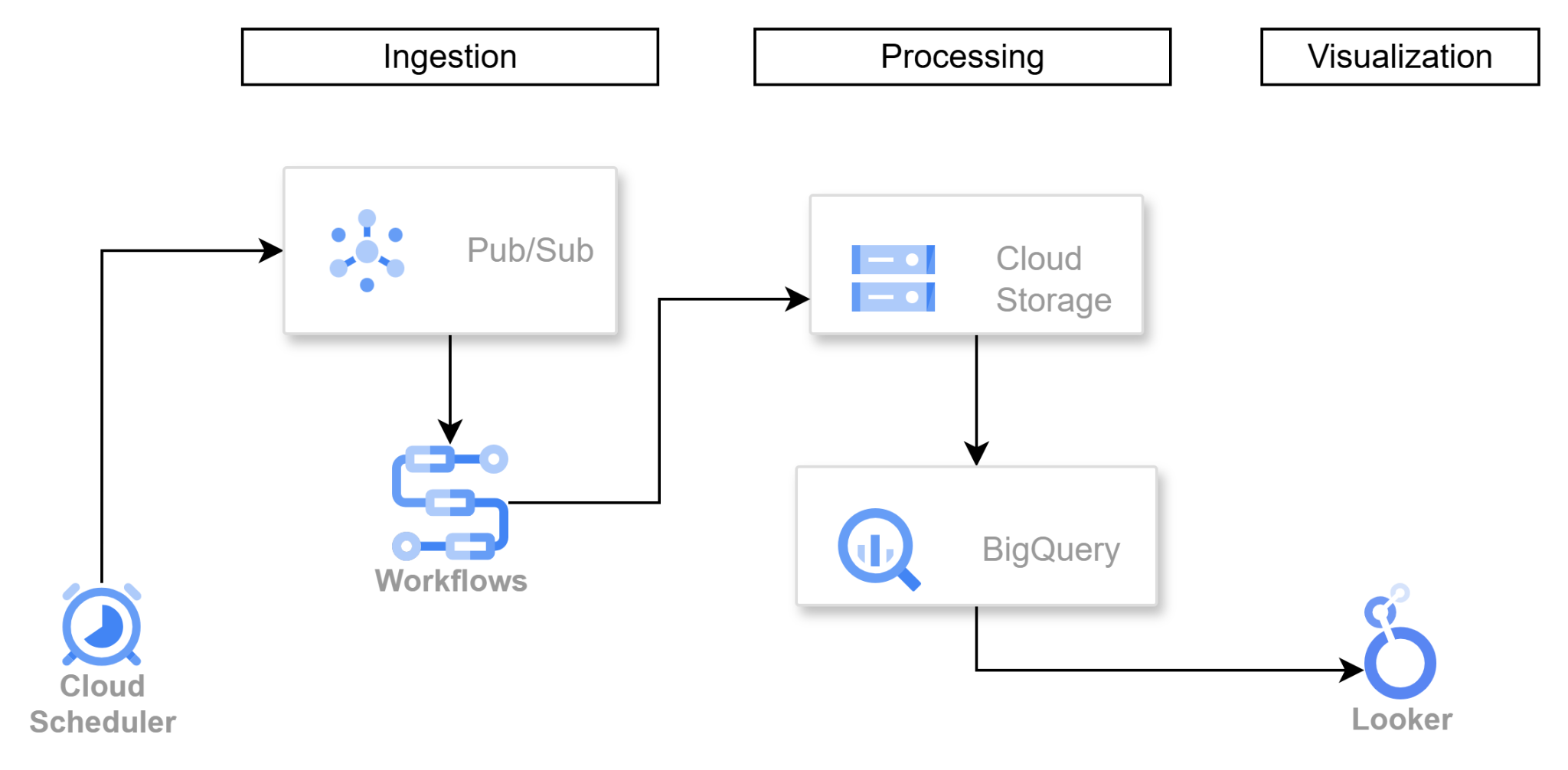
### **Security, Governance, and Protection**

* **IAM Roles**: Least-privilege access enforced (Pub/Sub publisher for GCS, BigQuery Data Editor for Workflow service account).
* **Data Encryption**: All data encrypted at rest and in transit by default in GCP.
* **Audit Logging**: Enable Cloud Audit Logs for traceability.
* **Governance**: Separate storage buckets for raw and processed data; use data retention policies in GCS.

### **Data Processing Frequency**

* **Ingestion**: Weekly or monthly via Cloud Scheduler.
* **Processing**: Triggered by Workflow on each upload
* **Aggregation**: Performed via BigQuery view logic
* **Delivery**: Dashboard reflects latest data automatically (real-time BigQuery sync)

### **Visual Architecture Description**



*Image created through* [*draw.io*](http://draw.io) *software*

### **Advantages of This Conceptual Architecture**

* **Cost-effective**: All services used are within the GCP free tier.
* **Reproducible**: Code and infrastructure managed via GitHub; can be deployed using Terraform or gcloud CLI (optional).
* **Modular**: Each component is independent and easy to maintain or upgrade.