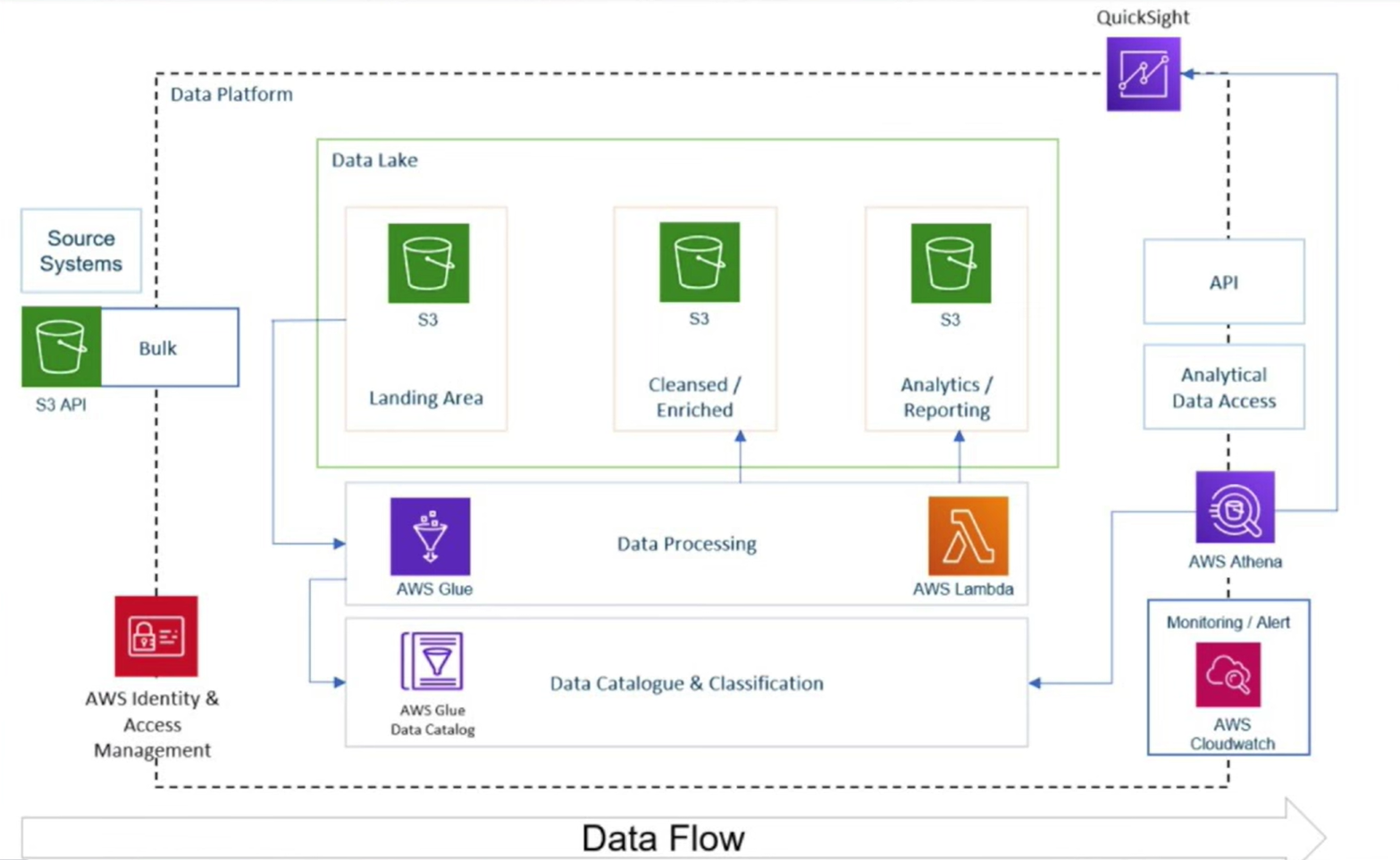
End – To – End DE project

Data Engineering YouTube Analysis Project by Martina Apostoloska

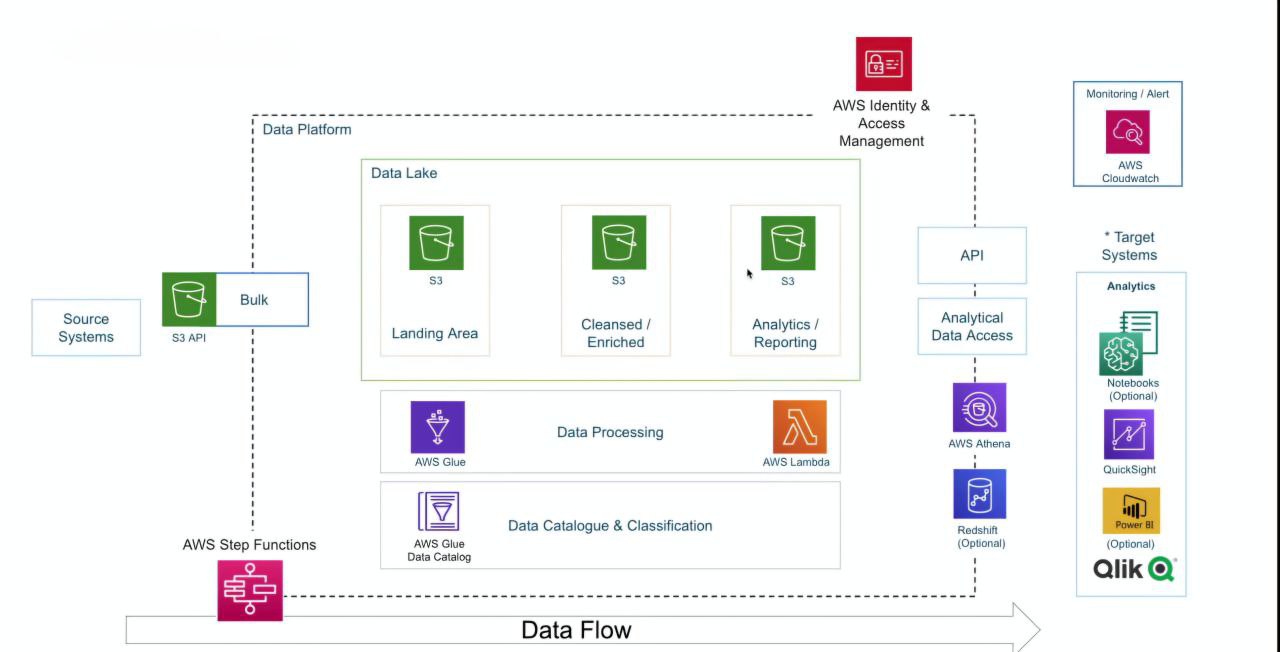
* Project architecture:  
  

*Landing area means where we uploaded our data,*

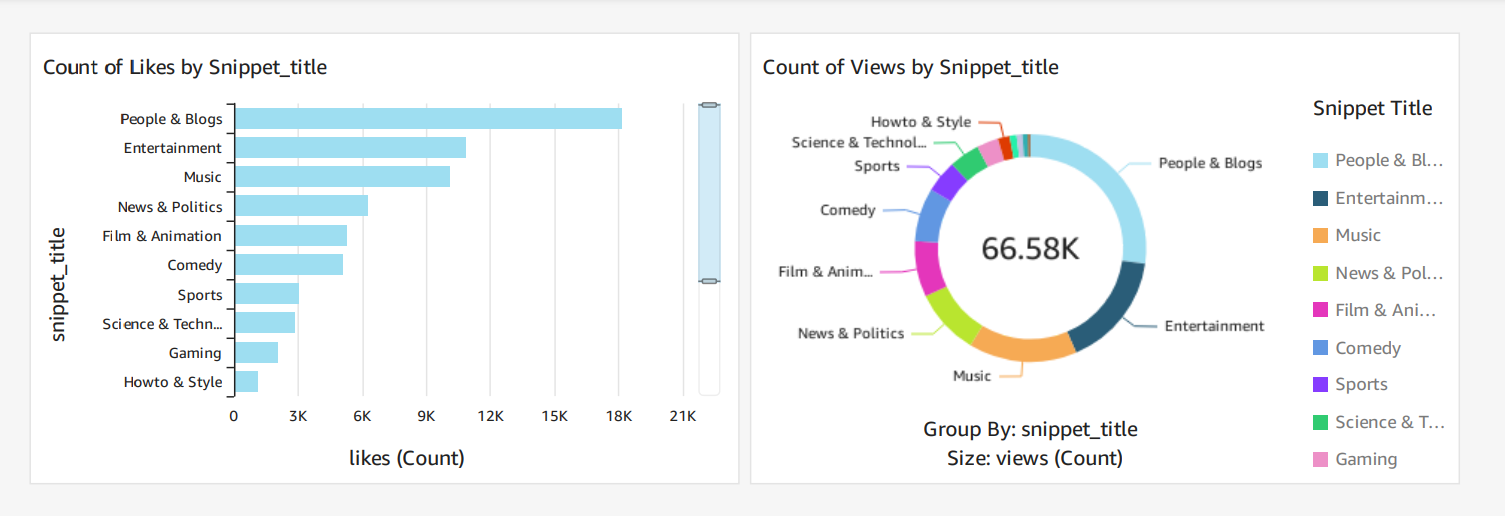
* **Overview:**   
  This project demonstrates an end-to-end data engineering solution designed to manage, streamline, and analyze structured and semi-structured YouTube data based on video categories and trending metrics. The project was executed entirely on AWS to leverage the scalability and flexibility of cloud computing, focusing on efficient data ingestion, transformation, storage, and analysis   
  *Scalability*: Ensured the architecture could scale as the data size grows by leveraging AWS cloud services.

**AWS Services Used:**

* **Data Ingestion**: Built a robust mechanism to ingest data from multiple sources (CSV files) using Amazon S3.
* **ETL System:** Transformed raw YouTube data into a clean, analyzable format using AWS Glue and Athena.
* **AWS IAM**: Managed secure access to AWS services and resources.
* **Data Lake**: Centralized the data from various sources into a unified storage solution using Amazon S3 as a data lake.
* **Cloud-Based Processing**: Used AWS cloud infrastructure to handle large datasets that couldn't be processed locally.
* For **cataloging** I used Glue, AWS Glue
* **AWS Lambda**: Leveraged serverless compute to automate certain parts of the workflow.
* **AWS Athena**: Used for querying data directly from S3 using SQL, without needing to load it into a database.
* Source for data: <https://www.kaggle.com/datasets/datasnaek/youtube-new/data>



* **Reporting :Amazon QuickSight**: Developed interactive dashboards for visualizing trends and metrics.



**Key Takeaways**

* **End-to-End Solution**: Covered every aspect of data engineering, from ingestion and transformation to reporting.
* **Scalability**: Successfully handled large data volumes using cloud-based solutions.
* **Cloud Native**: Employed AWS services to process, store, and analyze data without relying on on-premise resources.