API Interface for Analytics on IMDB Data

Abhinivesh Palusa
Nikhil Prabhu

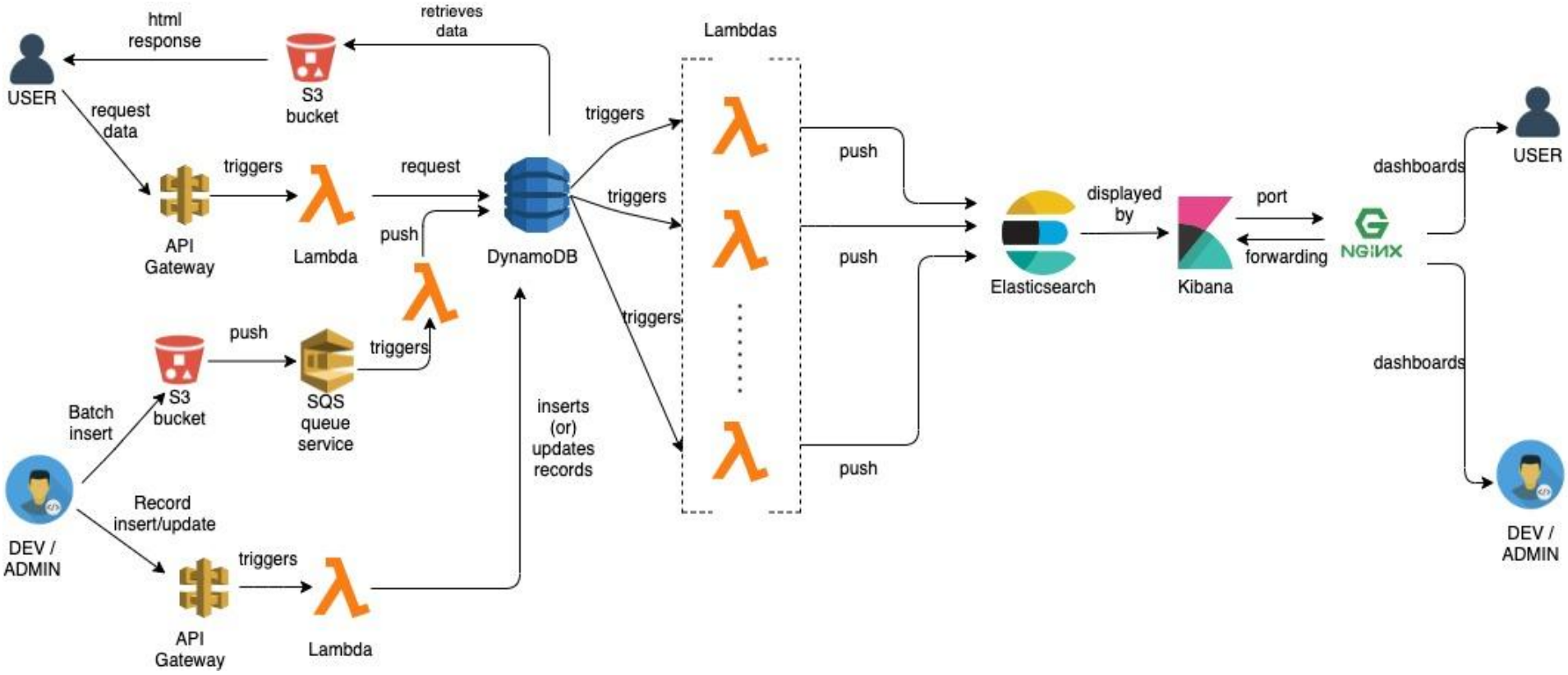
CSCI 5253
Project Submission

University of Colorado
Boulder

Purpose

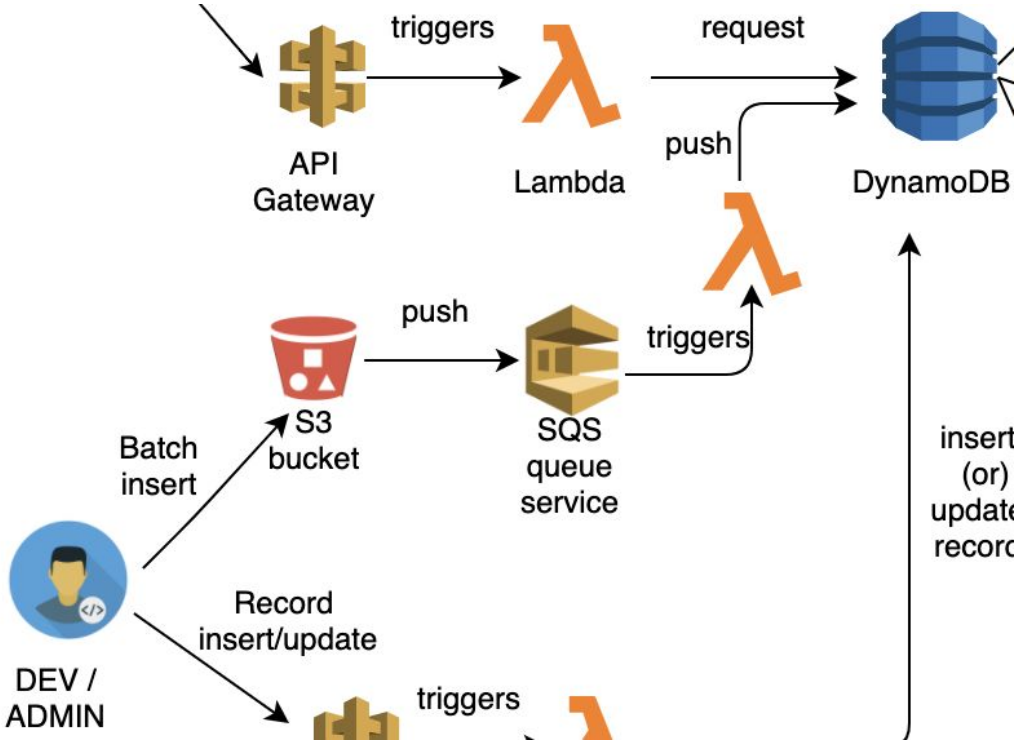
- Create an API (IMDB doesn’t have one) to provide analytics on IMDB data.
- Update analytics whenever an insertion, updation or deletion happens to any of the present records.
- An API to describe shows (titles) or describe people working in those shows.

Overview/Architecture



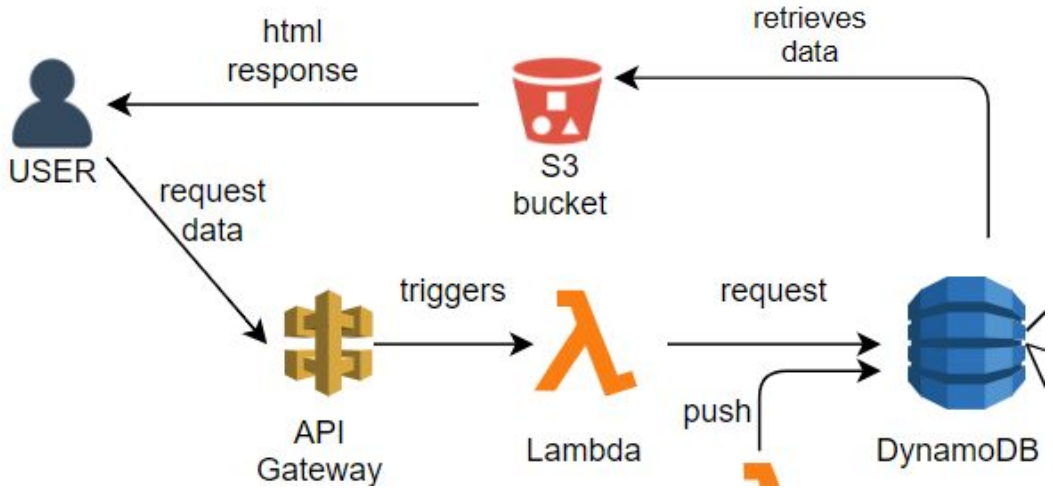
Flow 1:
S3 → SQS → DynamoDB

- Batch of data falls into S3 pushed by developer and λ is triggered.
- This λ pushes data record-wise into SQS and one more λ is triggered.
- The second λ pushes each message/record into relevant table of DynamoDB.



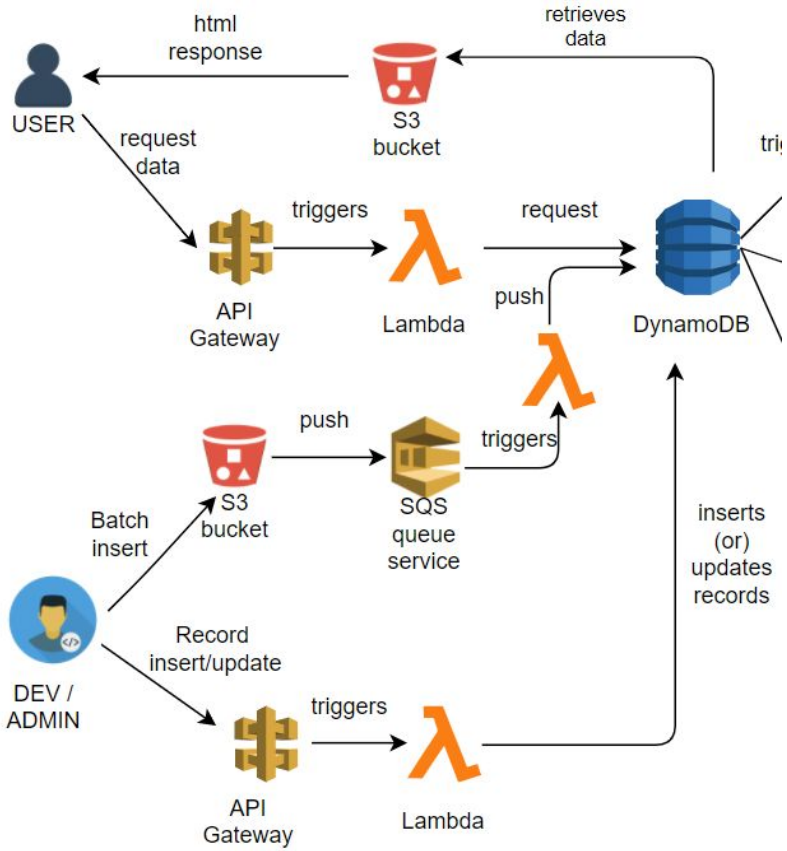
Flow 2:
API Gateway → DynamoDB → S3

- User requests data via API Gateway triggering a λ to a DynamoDB database.
- This λ transforms data to a relevant format collecting data from required tables in the DB.
- This data is stored in a HTML document in S3 whose public URL is sent as response.



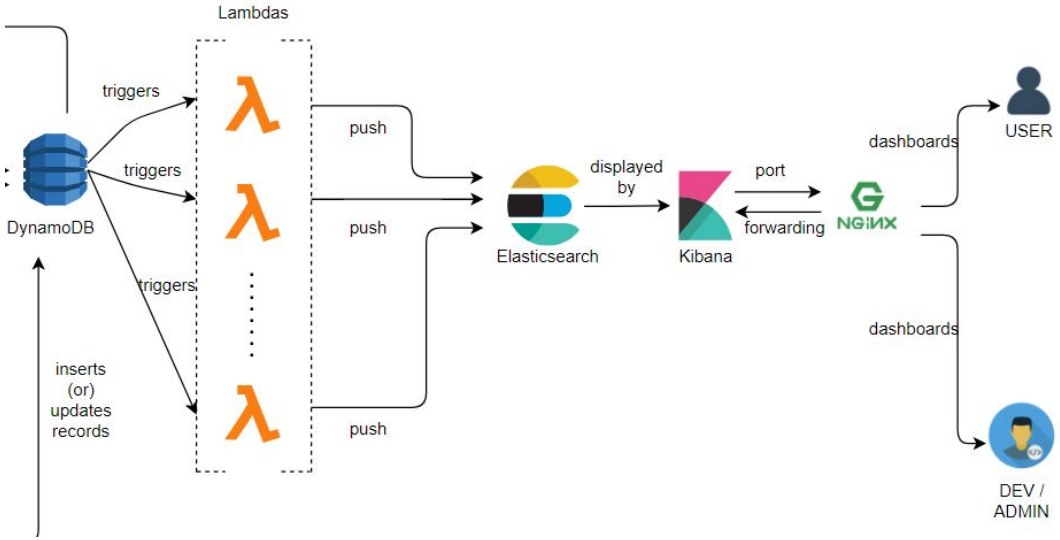
Flow 3:
API Gateway → DynamoDB

- A record to be inserted / updated is posted via an API Gateway triggering a λ.
- This λ updates the relevant record or inserts a new one in the corresponding table of the DynamoDB database.



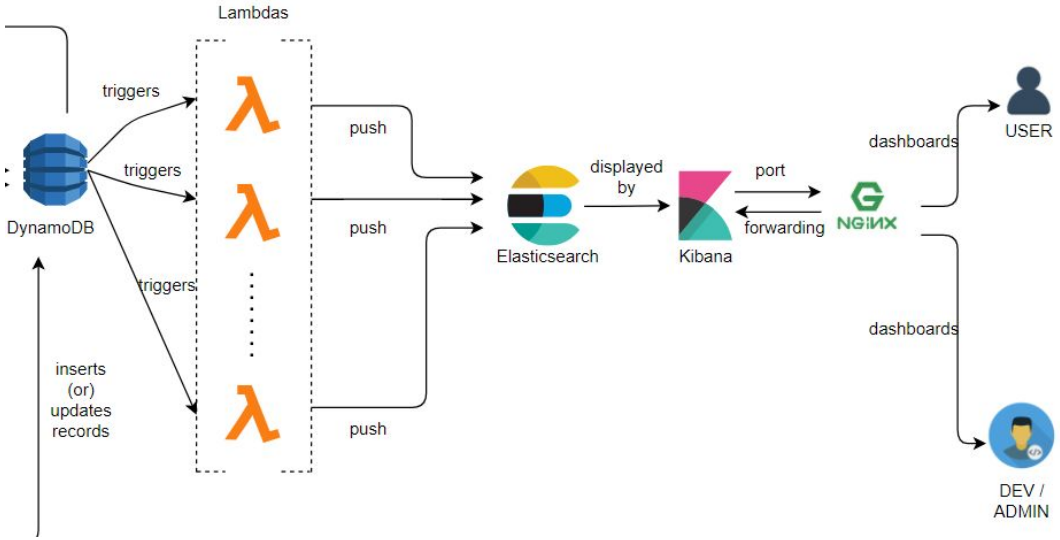
Flow 4:
DynamoDB → ES → Kibana [NGINX]

- Whenever data is inserted or updated in DynamoDB, a λ is triggered.
- This λ transforms data into a relevant JSON doc to be pushed into one of the available two indices on Elasticsearch (titles and people).



Flow 4:
DynamoDB ⟶ ES⟶ Kibana [NGINX]

- These two indices can create many visualizations on Kibana.
- Kibana port 5601 is forwarded from port 80 by NGINX.



Testing and Debugging

- Designed classes for each service (S3, SQS, DynamoDB, Elasticsearch) for better development and testing purposes.
- Created unit-tests around above services and API Gateway using a mini-batch of data (5 records from each table with 7 tables).
- Integration tests for each of the 4 flows using same type of mini-batch.

Testing and Debugging

- Integration tests for the entire architecture of the 4 flows.
- Debugging data formats between components using Cloudwatch logs.
- Using Kibana Dev tools to check whether the data is inserted in the expected format using various APIs like Insert API, Delete API, Count API, Bulk API, etc.

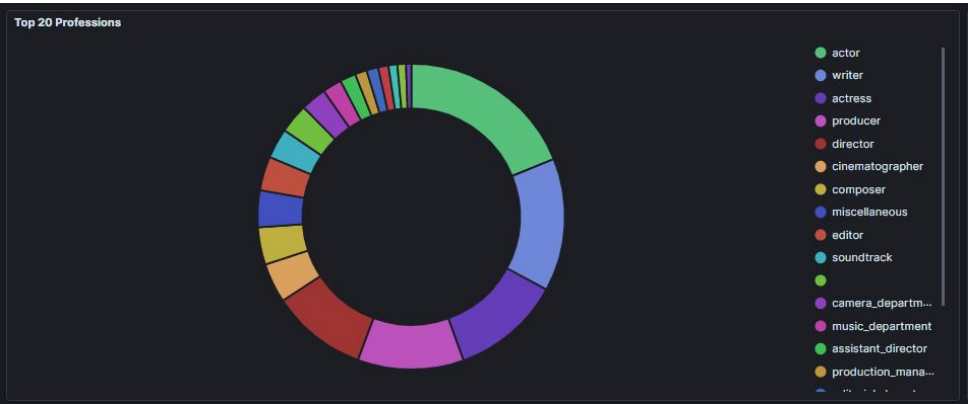
Limitations

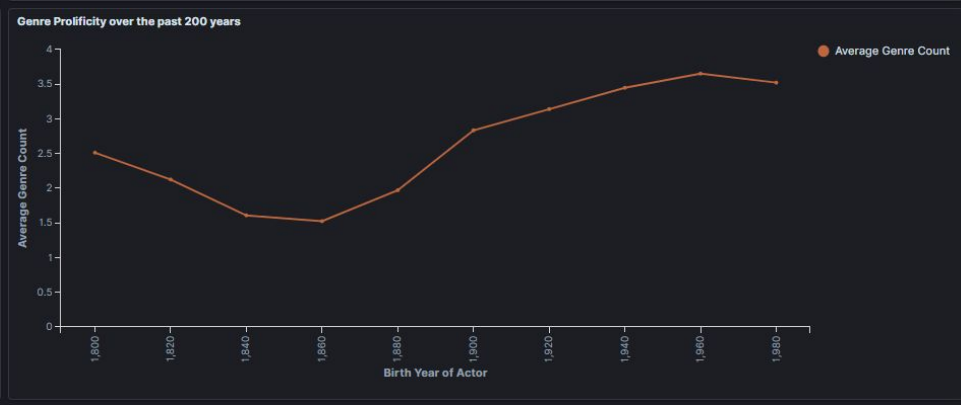
- Limited AWS credits (100) and limited time per session (3 hours).
- Couldn’t store the entire 10GB of data and couldn’t do batch upload for all tables in any of the AWS services and Elasticsearch on EC2 instance. Only one table at max per session.

Future Work

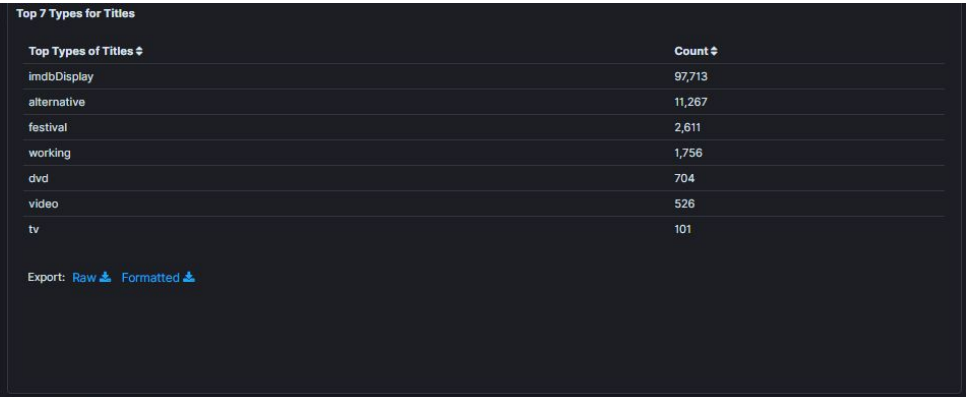
- Performing advanced data transformations while moving data between components using corresponding λ functions.
- More number of visualizations on the Kibana dashboard.
- Currently limiting data to ‘en’ language and can be extended all available languages on IMDB.
- Services involving container architecture (Docker & K8s).

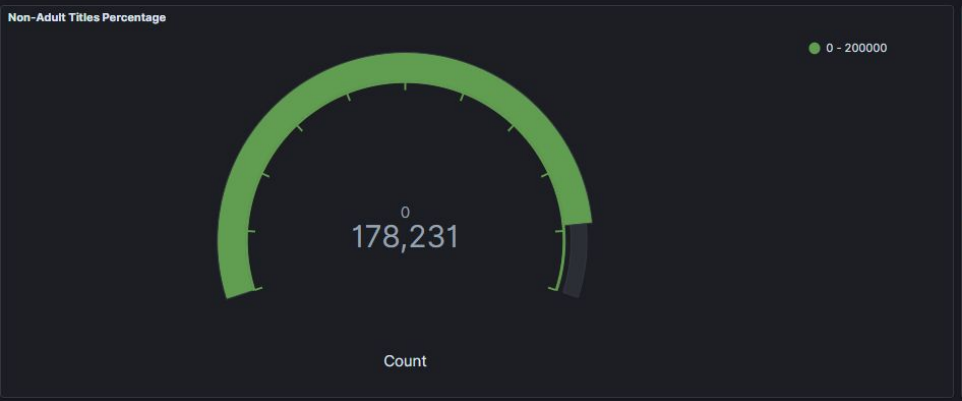
Dashboard (People)

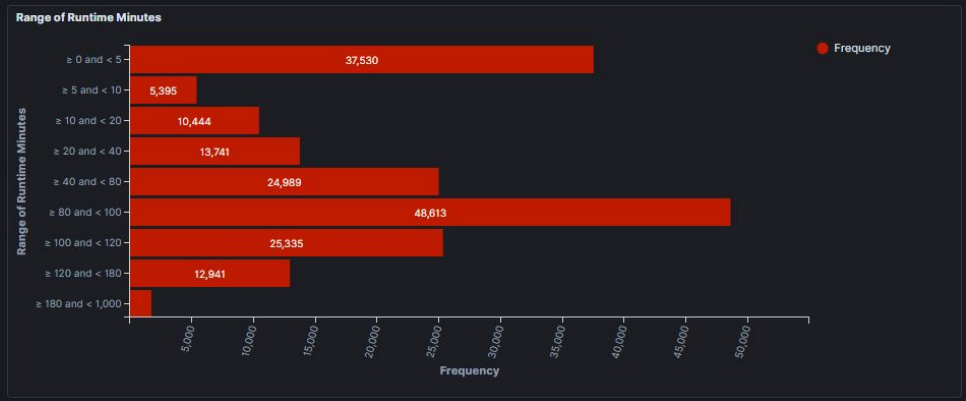




Dashboard (Titles)







THANK YOU