Name of the customer (Internal or affiliate case studies will not be accepted)
Comafi Bank

AWS Account ID (Will be used to verify AWS service usage)
008729895921

Problem statement/definition

Comafi Bank is a private capital bank located in Buenos Aires city, with 65 sites distributed in capital cities of Argentina. It has no sites on foreign countries.

The bank was on the verge of releasing a series of onboarding digital processes which involved web applications and tailored business circuits and tasks using aws step functions, lambda, api gateway, dynamodb and AWS Elastic Search. (Onboarding Application onwards).

The Onboarding application is a serverless application developed in NodeJS and entirely hosted inside AWS services. With this in mind, the customer contacted us in order to develop an ETL process from the logs and app data, to process, aggregate and leverage this information. The full requirement included the implementation of a cloud data lake and an analytics platform that could store and centralize the business application and logging data.

Additionally, there was a requirement that highly variable data needed to be stored in a data warehouse-like service, using a tabular format, so their BI area, which mostly has SQL knowledge, could easily create operations and high level reports using Power BI. Power Bi was a constraint due to pre existent licensing deals that the customer had, locking that reporting tool for some time.

Therefore our tasks during this project were:

1. Implement a data lake structure supporting application and logging data
2. Process and store the aforementioned data, through ETL processes.
3. Store additional needed data in a tabular format, in a data warehouse, so it could be leveraged by the BI area.
   There was also a time and budget constraint.

What you proposed

In order to give an answer to the requirement, we proposed the use of AWS services as cloud infrastructure to support the data lake and analytics platform. As implementation time was an important constraint, we suggested doing an iterative process through different MVPs, in order to be able to respond to the immediate requirements as fast as possible, and improve the workflow afterwards.

With this in mind, the initial suggested architecture included the use of:

- Redshift as a Data Warehouse to support fast OLAP
- S3 to support the Data Lake structure
- AWS Glue for the ETL process and for crawlers.
- Athena to query over S3 data and to be used as connector for PowerBI.

How AWS services were used as part of the solution

The following services were additionally used, in addition to the ones previously mentioned:

- Lambdas were used for serverless processing and events management.
- AWS KMS was used to support encryption keys.
- CloudWatch was used to hold and check logs.
- SQS was used to orchestrate process and reduce the overload on redshift writes.
- SNS was used to manage notifications and email sending.

Third party applications or solutions used

PowerBI

Start and end dates of project (Case studies must be for projects started within the past 24 months, and must be for projects that are in production)

July 2019 to February 2020.

Outcome(s)/results

The initially suggested architecture received several updates due to budget constraints and additional client restrictions. This difficulties were taken into account and we managed to create a successful solution that met the client requirements while trying to find the best way to introduce the architecture best practices.

The implementation of the data analytics zone, including a data lake in three zones, a Data Warehouse supported by Redshift, and the automation of the ETL processes, allowed the data & analytics team at Comafi to improve their productivity, allowing them to work in isolated environments in order to have a more flexible area to modify data without disrupting the business processes, and a faster and wider access to existing data.

These advantages allowed Comafi to better leverage their data, improving their decision making process and obtaining faster conclusions through the optimized cloud infrastructure that, now, supports their processes.

Lessons Learned

Every solution has to be tailored to the client needs, even though the first approach is to follow the best architectural practices. Budget and time constraints are a challenge that needs to be taken into account when developing the solution.

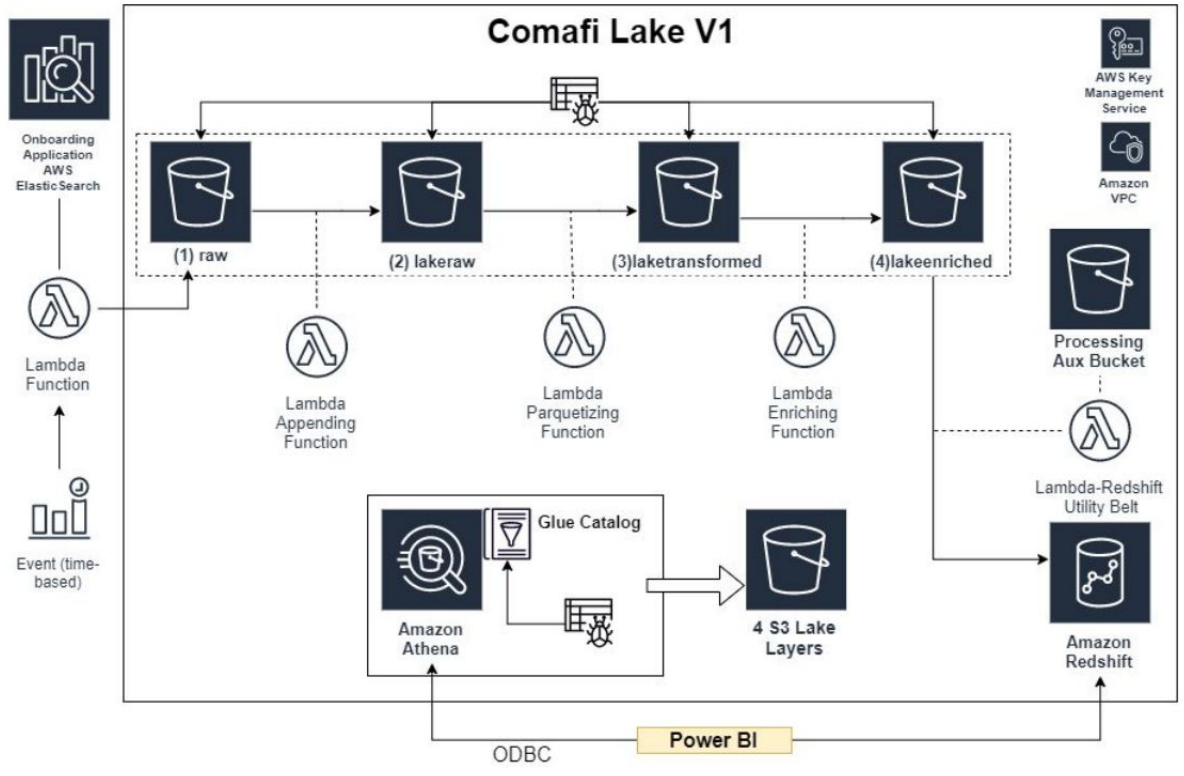
2.2Architecture Diagrams

The initial diagram proposed consist, as we mentioned, in the use of S3 to support the data lake, redshift to support the data warehouse and glue jobs to perform the ETLs processes. In this case, since all the processes were directly hosted inside AWS (both the onboarding application and the whole ETL process, storage and reporting), everything was managed through private networks. There is no external traffic.

The selected region is North Virginia (us-east-1), with subnets ranging through different Availability Zones. The CIDRs used were:

- 172.26.64.0/26
- 172.26.64.64/26
- 172.26.64.128//26
- 192.168.96.0/26
- 192.168.96.64/26
- 192.168.96.128/26

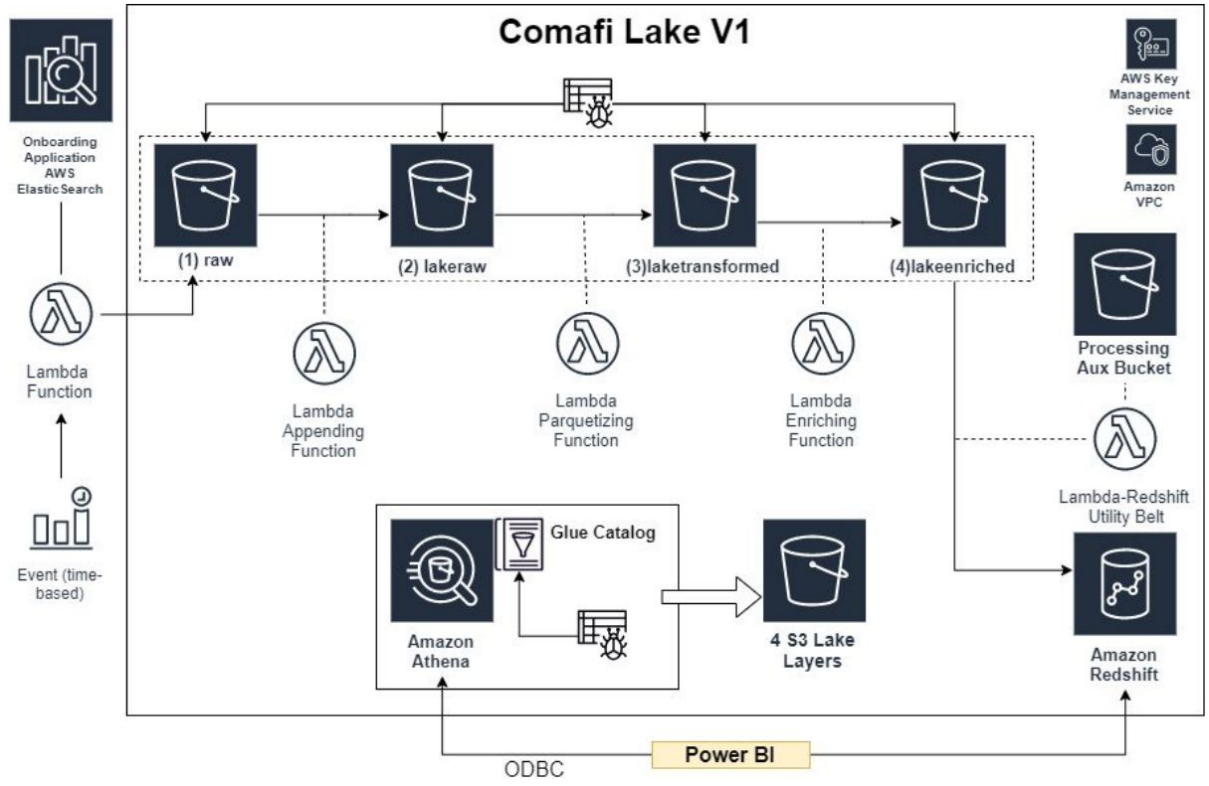
The solution proposed at first was:



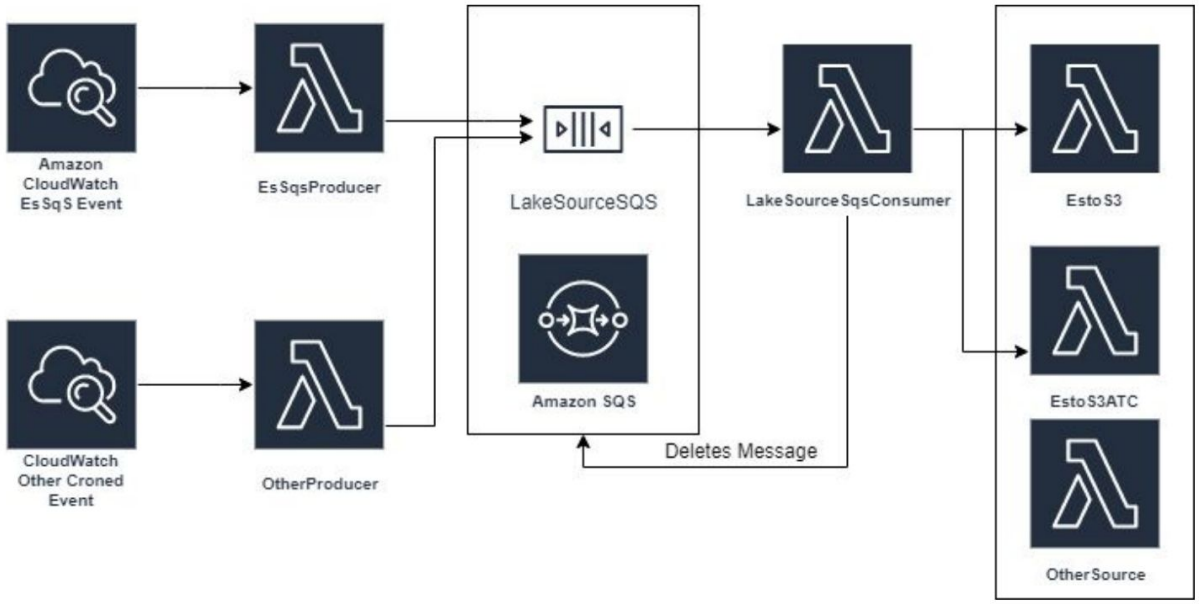
During the development process, the client required to reduce glue costs and achieve a faster processing time. Since the ETL processes carried out were relatively small, we migrated glue jobs to lambdas. Additionally, in order to synchronize the different lambdas related to the process we proposed to use SQS.

The resulting architecture is depicted in the following diagrams.

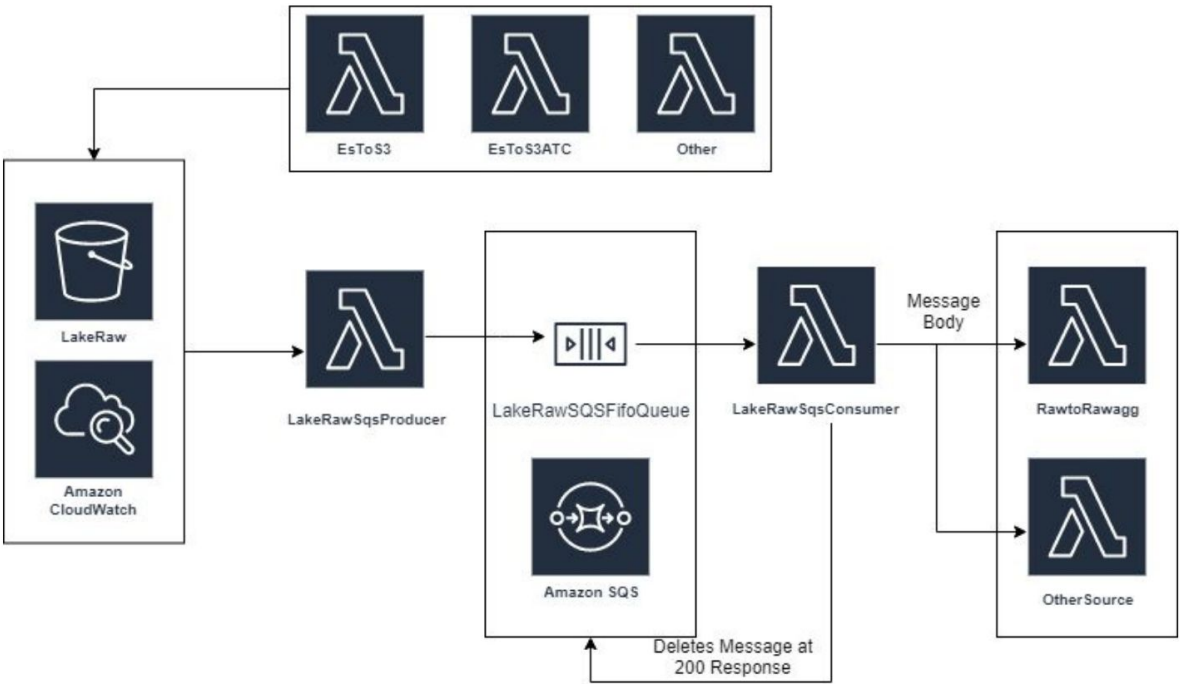
General Diagram:



Use of SQS to orchestrate the data flow between processes. Layer 1



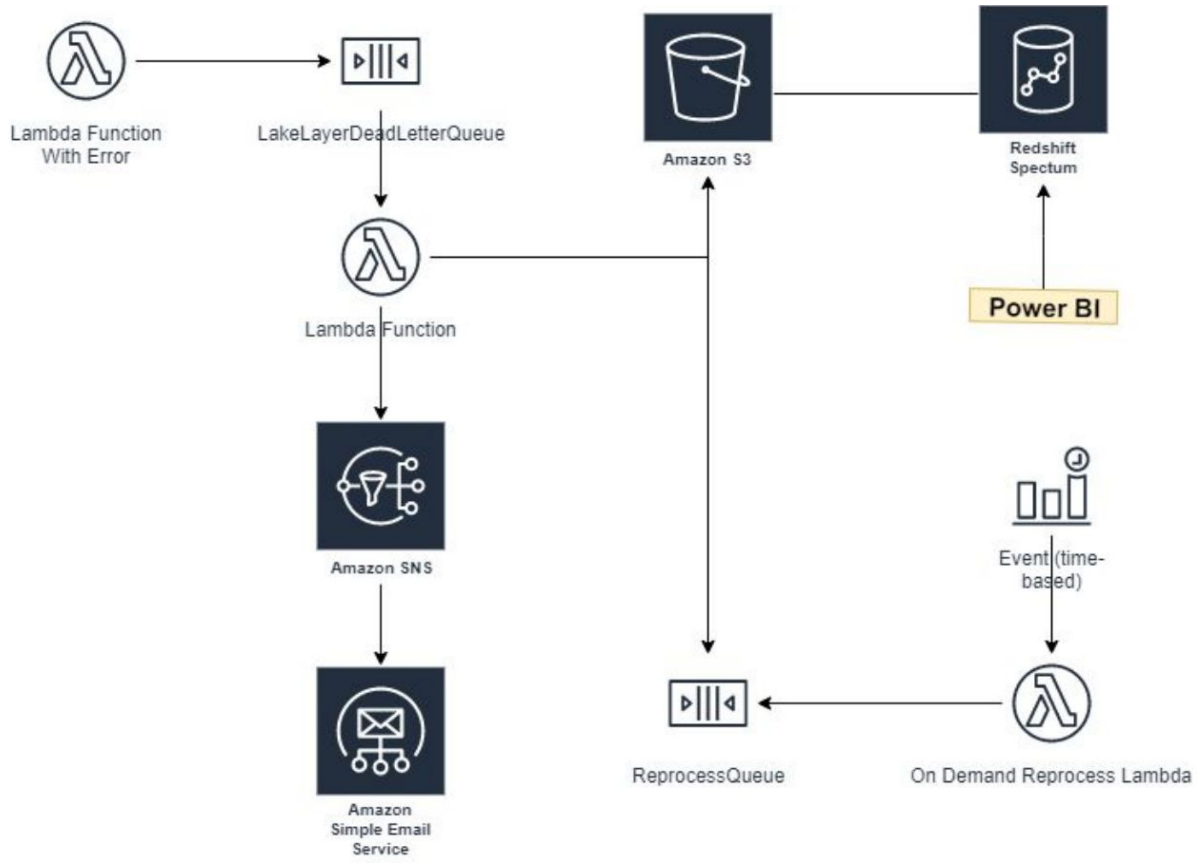
Use of SQS to orchestrate the data flow between processes after writing to S3. Layer 2



Monitoring Solution. Layer 3

Any error produced ends up in a SQS Dead Letter Queue, independent for each layer transition, mapped to the normal execution sqs queue. Every single queue is linked through triggers to a general handle lambda which persists the message in s3 bucket and does a little transformation to leave in a friendly readable tabular format which can be queried using Athena or Redshift Spectrum.

Additionally, an email is published through an SNS and most important, the error messages with all the data necessary for reprocessing go to a max retention period configured sqs where an on demand reprocessing on demand can be executed.


Lambda Function
With Error
LakeLayerDeadLetterQueue
Amazon S3
Redshift
Spectum
Lambda Function
Power BI
Amazon SNS
Event (time-
based)
Amazon
Simple Email
Service
ReprocessQueue
On Demand Reprocess Lambda

RED-001 - Data storage is optimized based on metrics and patterns

In order to choose sorting keys, we had several meetings with the different stakeholders to check how data needed to be served. This meetings allowed us to better understand the queries that were going to be made and, thus, choose the appropriate sorting keys for each case.

The data structure stored by the Redshift DW for Comafi consists in a enriched log in a tabular format, that reports certain aspects of the onboarding application developed by Comafi, which is used by their customers. The queries to be made on this data are, mainly, related to log times. The BI analysts will be querying data with two approaches. The first approach is to query data to obtain the most recent logs. The second one is filtering on a time range (weekly, monthly). The sorting keys, therefore, were timestamp based.

Compression was managed by Redshift automatically in the following ways:

- Columns that are defined as sort keys are assigned RAW compression.
- Columns that are defined as BOOLEAN, REAL, or DOUBLE PRECISION data types are assigned RAW compression.
- Columns that are defined as SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP, or TIMESTAMPTZ data types are assigned AZ64 compression.
- Columns that are defined as CHAR or VARCHAR data types are assigned LZO compression..

The maintenance of the Redshift cluster includes regular vacuuming, according to the frequency and quantity of data loaded. For this particular case, and according to the work hours and peak consumption we suggested a nightly schema.

RED-002 - Redshift Database User Access Management and Security is following best practices

The AWS account was set up from scratch, the root account remained in the hands of the customer, and we created an IAM user for the engineer deploying the infrastructure with it's proper permissions and an IAM group for the customer operations team. Specifically for Redshift, the following roles were created:

- A role for services and applications that would query the data in Redshift (like PowerBI), only giving permissions to list and read data.
- A role for Lambdas which wrote data to Redshift were given write and create permissions, according to the tables needed.
- A role for Redshift that also has list and read permissions for S3 in order to use Redshift Spectrum to read data stored in the data lake.

Finally, data in Redshift was encrypted with KMS service.

RED-003 - Workload Management is configured properly to meet application needs

When we migrated from glue jobs to lambdas, which allowed us to provide a faster workflow at a significantly lower cost, we started having issues with workload in the redshift

master. The problems were related to an over saturation of COPY operations. We tried to apply a custom workload management, and we even tested upscaling the number of nodes available. However, we didn't achieve the expected result, since cost constraints were really tight. Therefore, we managed to implement a schema where we buffered data to be written to the RedShift cluster through a SQS queue. This allowed us to meet the client requirements both in cost and in operational speed. The final implementation was a schema with (1)Producer - (2)Queue - (3) Producer - (4) Lambda Transformation between every layer.

RED-004 - Solution Composition Requirements

In this case we built a data lake, supported by Amazon S3, which is queried both by Athena and by Redshift Spectrum. The data lake is partitioned on a date schema, according to the client needs, in order to reduce the amount of data scanned in each query. Data on Redshift is also partitioned and sorted into a datetime schema, in accordance to the needs of the business layer of Comafi. Data Modeling was done in collaborative work with the team at Comafi, and was built upon their needs, respecting AWS best practices where possible.

General

ACCT-001 - The root user is secured

Root User has not been assigned access keys and has been only used to manage the first accounts, admins and power users. MFA is enabled on the root user.
Each user has been assigned a user, and services that interact with other services have been assigned roles. Policies and group permissions have been elaborated with an acceptable and secure granularity.

ACCT-002 - Account contact information is set

The Operations, Billing, and Security contact email addresses are set, and all account contact information, including the root user email address, is set to a corporate email address.

ACCT-003 - AWS CloudTrail is enabled

AWS Cloudtrail was suggested as an important part of the Comafi architecture. We implemented this feature with a log of the cloudtrail events dumped to an S3 specific bucket.

Operational Excellence
Requirements in this category relate to the ability of the APN Partner and the customer to run and monitor systems to delivery business value and to continually improve supporting processes and procedures.

OPE-001 - Metrics are defined for understanding the health of the workload

In order to better understand the health of the workload, both cloudwatch and cloudtrail logs were analyzed. We designed metrics that allowed to assess the number of lambda functions that were successful, the number that failed, most common fails, times consumed by each process, amount of data processed and stored by each process, and services up and down time, with the additional metrics of cpu and I/O utilization, throwing alerts on peaks.

OPE-002 - Workload health metrics are collected and analyzed

As stated before, the metrics are collected through cloudwatch and cloudtrail and stored in an s3 bucket for further analysis. This analysis is crawled by glue and queried through athena, allowing it to be visualized with the PowerBI license the client currently owns. We have recommended the use of AWS QuickSight in order to facilitate the reporting of these metrics. The AWS Personal Health Dashboard is also used to monitor certain workload health metrics, due to the easiness of use.

OPE-003 - Operational enablement

The handover process was carried out by stages. In this line, we handled documentation and accompanied the operational personnel understood the code developed and the solution as a whole. The IT personnel, who will support the solution, were familiar with AWS cloud services, since they already manage other solutions. We worked with possible fails with lambdas, SQS and redshift, showing them how to debug the errors through cloudwatch logs and AWS services, including code fails, health checks and fails, and errors due to insufficient concurrency.

OPE-004 - Deployment testing and validation

A Development / Production schema was used to test the correct implementation of every part of the process, including ingestion and etl jobs. Development was carried out and, later on, after testing occurs in the development environment, the development team merges work to the production environment and tests are done in the new environment again. This schema was carried out through out the development process, and certain automation was done through cloudformation

We handled over the process to Comafi IT teams once we reached production stage for the whole process, training them in the use, monitoring and further development in case they wanted to add new functionalities.

OPE-005 - Code assets are version controlled

The code is being version controlled through the use of git and github as cloud repository. As part of this process, certain aspects of the process were handled as infrastructure as code and versioned in the same repositories, thus allowing CI/CD in these aspects.

OPS-006 - Application and workload telemetry

Lambdas and SQS log execution data to Cloudwatch, as well as Redshift. Lambdas logs include details of the execution process in order to be able to easily debug the data processes. Metrics about concurrency and usage are also logged and can be queried if needed.

Security - Identity and Access Management
Requirements in this category focus on best practices around AWS Identity and Access Management (IAM) and other identity and access management systems owned by the APN Partner.

IAM-001 - Access requirements are defined
We received an account with certain limitation in order to carry out only what needed. We created and assigned roles and users as needed, looking to maintain the permissions as limited as possible.

IAM-002 - Grant least privileges
Policies created were given fine grained access to the services and permissions needed. As an example, we submit an example policy to unload data from Redshift to S3:

```
{
  “Version”:”2012-10-17″,
  “Statement”:[
    {
      “Effect”:”Allow”,
      “Action”:[
        “s3:PutObject”,
        “s3:DeleteObject”,
        “s3:ListBucket”
      ],
      “Resource”:[
        “arn:aws:s3:::redshift-testing-comafi*”
      ]
    }
  ]
}
```

IAM-003 - Static AWS Access Keys are not used for programmatic access.

There are no processes that use programmatic access through AWS Access Keys.

IAM-004 - Unique non-root credentials are used for interactive access.
Each consultant access the account with a different IAM user and credential. These are provided by the IT Security Department of Comafi. Credentials were not shared between individuals.

Security - Networking
Requirements in this category focus on security best practices for Amazon VPC and other network security considerations.

NETSEC-001 - Security groups are tightly scoped.

Security groups were restricted to the minimum access policies, only allowing the needed ips and ports.

NETSEC-002 - Data that traverses the Internet is encrypted in transit.

There are no endpoints traversing internet.

NETSEC-003 - Data stores are in private subnets.

For storage, we used both S3 and Redshift. Both data stores are in private subnets (S3 is accessed through a VPC endpoint).

Security - IT Operations
Requirements in this category focus on IT security operations best practices including logging, monitoring, incident response, and data classification.

SECOPS-001 - Cryptographic keys are managed securely.

Data is encrypted through the managed AWS services. No user provided keys were used.

AWSAPI-001 - Official AWS SDKs are used to call AWS API endpoints.

Programming Language used was python and pyspark. The access to aws services were done vía AWS Cli, and AWS Wrangler was used in the ETL process.

Reliability
Requirements in this section focus on the ability of the solution to prevent, and quickly recover from failures to meet business and customer demand.

REL-001 - Deployment automation.

In order to deploy services, during the development process, AWS Management Console was sometimes used, due to the simplicity. Once a stable phase was achieved, further changes and solutions were done through AWS CLI and Cloud Formation.

REL-002 - Availability requirements are defined for the solution.

In relation to individual zone failure, RTO and RPO is automatically managed by AWS since it uses replication and continuous backups to enhance availability and improve data durability and can automatically recover from node and component failures.
In case of availability zone disruption, the RTO and RPO is managed through snapshots storage, with a periodicity according to the requirements set by the client. In case of a availability zone disruption, the RTO and RPO max is around 2 hs, with the process taking into account the restoration of the snapshot saved in S3.

REL-003 - The solution adapts to changes in demand.

The provided solution adapts to changes in demand. ETL jobs are carried out mostly through Lambdas, that will be concurrently launched as more demand is needed. In the same sense, S3 allows scalability in a transparent way. Redshift nodes where not set on auto scaling since the client wanted to manage them manually in order to keep the cost as tight as possible.

Cost Optimization
Requirements in this category relate to the APN Partner's ability to help customers run systems that deliver business value at the lowest price point.

COST-001 - Total cost of ownership (TCO) analysis or cost modeling was done.

Through the process of development, we suggested several changes to the architecture, as can be seen in the different architecture diagrams. Some of these changes were carried out in order to optimize the costs of the solution The estimated costs for redshift included 4 environments, the first three being sandbox, dev and test, each with two nodes, and the fourth being production with a 3 nodes schema.

US East (N. Virginia)	Amazon Redshift	0	365	4380	USD	(2 instances) dc2.large OnDemand
US East (N.	Amazon	0	365	4380	USD	(2 instances) dc2.large

Virginia)	Redshift					OnDemand
US East (N. Virginia)	Amazon Redshift	0	365	4380	USD	(2 instances) dc2.large OnDemand
US East (N. Virginia)	Amazon Redshift	0	547,5	6570	USD	(3 instances) dc2.large OnDemand
Total Costs			1642,5	19710		

This estimation was evaluated according to the client needs, although we recommended to purchase upfront instances when possible.