*Enrique Zetina*

*EPAM Systems Inc. | enrique\_zetina@epam.com*

*Technology domains*

Cloud

Table of Contents

[Deployment view 2](#_Toc165205615)

[Summary costs. 3](#_Toc165205616)

[Cloud Firestore 3](#_Toc165205617)

[Why Firestore? 3](#_Toc165205618)

[Collections 3](#_Toc165205619)

[SLAs 4](#_Toc165205620)

[Firestore pricing 4](#_Toc165205621)

[Google Kubernetes Engine 5](#_Toc165205622)

[Why GKE? 6](#_Toc165205623)

[Microservices 6](#_Toc165205624)

[SLAs 6](#_Toc165205625)

[GKE Pricing 7](#_Toc165205626)

[Google Pubsub 7](#_Toc165205627)

[Why Pubsub? 7](#_Toc165205628)

[SLAS 8](#_Toc165205629)

[Pubsub pricing 8](#_Toc165205630)

[Google Dataflow 8](#_Toc165205631)

[Why Dataflow? 8](#_Toc165205632)

[SLAs 9](#_Toc165205633)

[Dataflow pricing 9](#_Toc165205634)

[Google BigQuery 9](#_Toc165205635)

[Why BigQuery 9](#_Toc165205636)

[SLAs 10](#_Toc165205637)

[BigQuery pricing 10](#_Toc165205638)

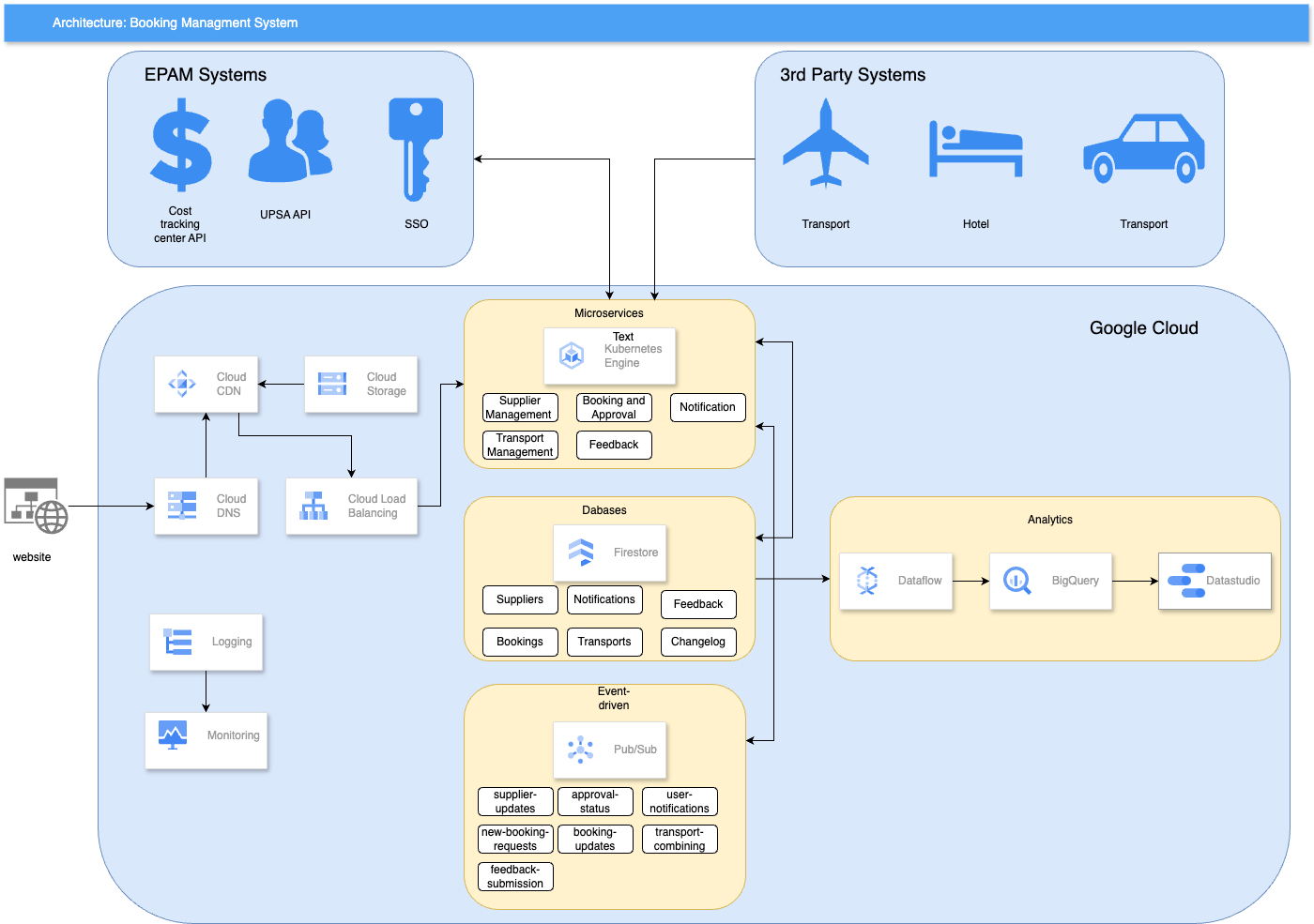
[Availability 10](#_Toc165205639)

[Operational Service 11](#_Toc165205640)

[Analytics Service 11](#_Toc165205641)

[Cost calculation 12](#_Toc165205642)

# Deployment view



## Summary costs.

### Total cost (18 months).

|  |  |  |  |
| --- | --- | --- | --- |
| **Environment** | **Cost Per Month (USD)** | **Duration (Months)** | **Total Cost (USD)** |
| Development | $497 | 18 | $8,951 |
| Staging | $746 | 10 | $7,459 |
| Production | $994 | 7 | $6,961 |
| **Total** |  |  | **$23,371** |

### Cumulative cost

A graph of growth in blue

Description automatically generated with medium confidence

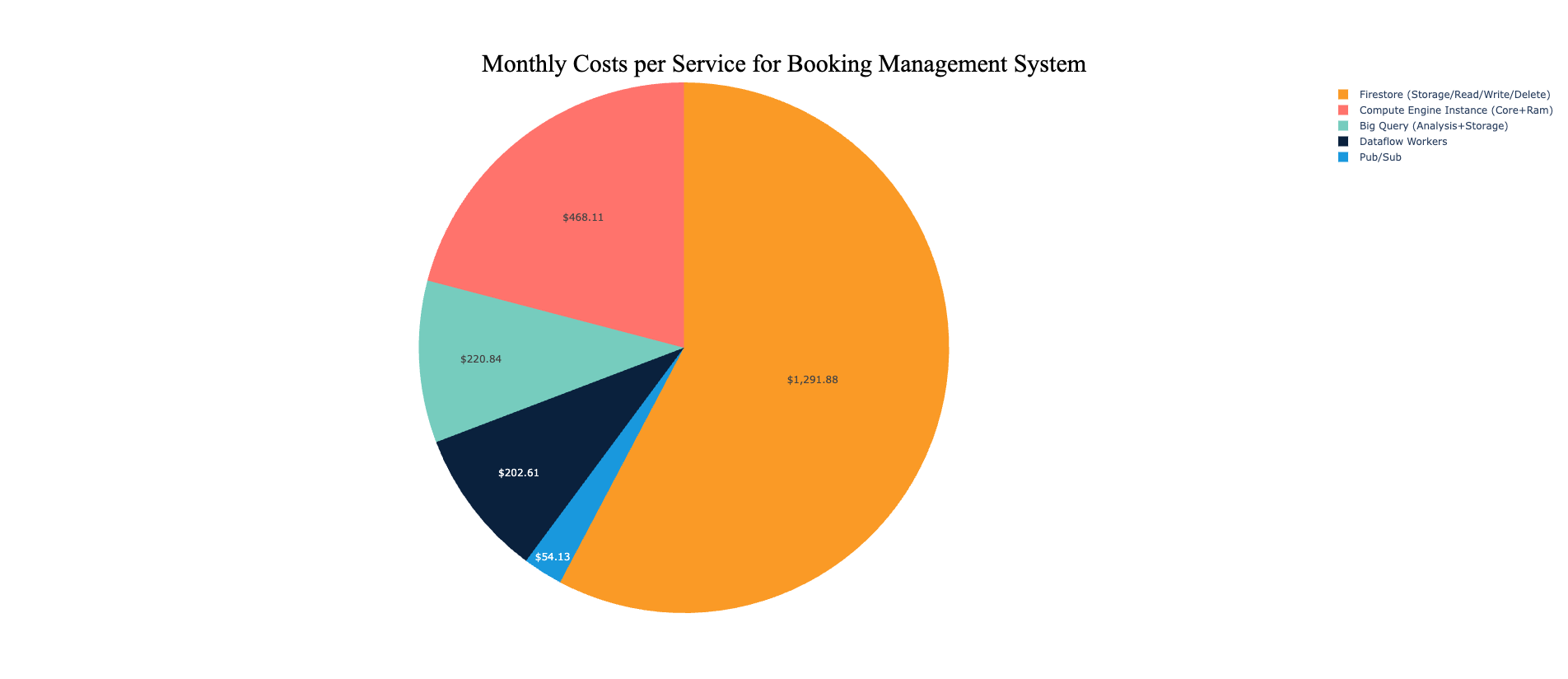
<https://github.com/ezetina86/Sas_Work_breakdown_structure-/blob/main/cumulative_costs.py>

Project timeline  
A graph with different colored bars

Description automatically generated  
  
https://github.com/ezetina86/Sas\_Work\_breakdown\_structure-/blob/main/gantt\_env.py

### Cost per month.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service Name** | **Development** | **Staging** | **Production** | **Total** |
| Cloud Firestore Storage | $287.09 | $430.63 | $574.17 | $1,291.88 |
| Pub/Sub | $12.03 | $18.05 | $24.06 | $54.14 |
| Big Query (Analysis+ Storage) | $49.08 | $73.61 | $98.15 | $220.84 |
| GKE | $104.03 | $156.04 | $208.05 | $468.11 |
| Dataflow Workers | $45.03 | $67.54 | $90.05 | $202.61 |
| **Total per month** | **$497** | **$746** | **$994** | **$2,238** |



<https://github.com/ezetina86/Sas_Work_breakdown_structure-/blob/main/service_cost.py>

### Cost per year.

|  |  |  |
| --- | --- | --- |
| **Environment** | **Cost Per Month (USD)** | **Cost per Year** |
| Development | $497 | **$5,967** |
| Staging | $746 | **$8,950** |
| Production | $994 | **$11,934** |
| **Total** | **$2,238** | **$26,851** |

## Cloud Firestore

Cloud Firestore is a NoSQL document database that lets you easily store, sync, and query data for your mobile and web apps - at global scale.

### Why Firestore?

Firestore is a NoSQL database that organizes data into collections of documents. Each document contains fields that map to values. This model is quite different compared to a traditional SQL database, where you typically have multiple separate tables. In Firestore, the data from different microservices can live in separate collections within the same Firestore database.

As for instances, Firestore automatically scales to match the workload, so it is not needed to manually create multiple instances.

### Collections

* Suppliers: This collection will hold the details of all suppliers (hotel and transport) including their name, contact information, available services, etc.
* Bookings: This will hold the details of all bookings made. Documents in this collection can include fields like booking ID, supplier ID, user ID, booking status, date, and time, etc.
* Users: The collection to store details of all users (employees and admins) including their personal details, contact information, and booking history.
* Hotels: This collection will contain details about the hotels including their location, availability of rooms, pricing details, facilities, and reviews.
* Transports: The collection to store details about transport options available including type of transport, availability, pricing details, and reviews.
* Notifications: This collection will hold information about all notifications sent to users like booking confirmation, cancelation, updates, etc.
* User\_Feedback: This collection can store feedback received from employees associated with specific bookings.
* Reports: A collection to store generated reports regarding booking details, supplier reports, and financial reports.
* Change\_Logs: A collection to trace back the changes made in booking or supplier details for audit purposes.

A collection may need to be further divided into subcollections based on the complexity of data and the relationships between them.

### [SLAs](https://cloud.google.com/firestore/sla)

|  |  |
| --- | --- |
| Covered Service | Monthly Uptime Percentage |
| Firestore Multi-Region | >= 99.999% |
| Firestore Regional | >= 99.99% |

### [Firestore pricing](https://cloud.google.com/firestore/pricing)

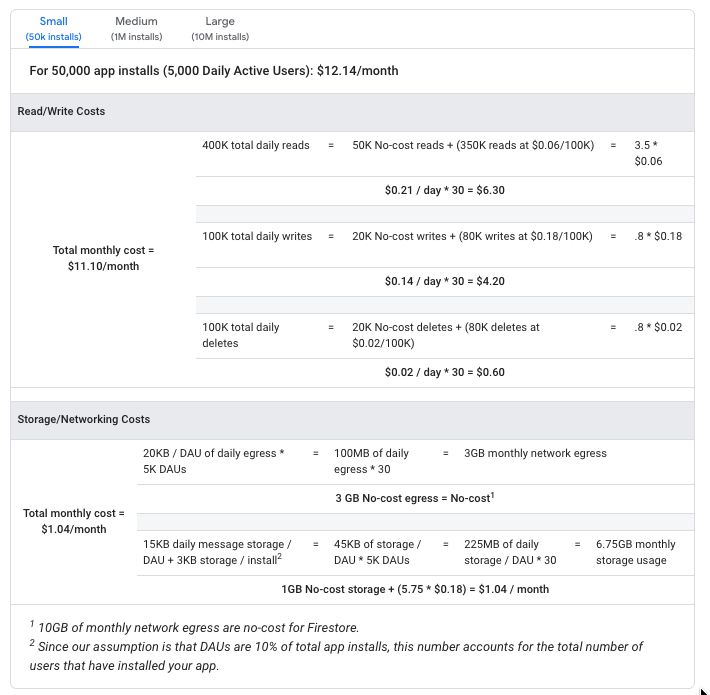
#### Free quota

Firestore offers free quota that allows you to get started with your (default) database at no cost. The free quota amounts are listed below.

Quotas are applied daily and reset around midnight Pacific time.

**Only the (default) database qualifies for the free quota.**

|  |  |
| --- | --- |
| Free tier | Quota |
| Stored data | 1 GiB |
| Document reads | 50,000 per day |
| Document writes | 20,000 per day |
| Document deletes | 20,000 per day |
| Outbound data transfer | 10 GiB per month |



## Google Kubernetes Engine

GKE is a feature-rich, managed Kubernetes platform that enables deployment, configuration, and orchestration of containers using Google Cloud infrastructure.

### Why GKE?

GKE has better integration with Google's Cloud Monitoring and Cloud Logging services.

Kubernetes offers a broad set of functionalities such as service discovery, secrets management, and advanced deployment strategies (like Blue/Green or Canary deployments).

Kubernetes has built-in mechanisms to automatically scale the application based on CPU utilization or custom metrics in your application.

### Microservices

* Supplier Management Service: This service will be responsible for managing and storing supplier details, handling search and prioritization logic for suppliers based on bookings and facilitating manual bookings by Travel Managers.
* Booking and Approval Service: Handles the creation and modification of bookings, executing various stages of the booking process such as proposed, accepted/rejected, and paid. Also handles booking cancellation and changes.
* User Management Service: Manages user details, logins, and roles. This may interact with Google Identity Platform.
* Notification Service: Handles the sending of notifications via email or SMS when booking statuses change. Could leverage external services or APIs to send the notifications.
* Change Management Service: Deals with updating booking changes and merges any conflicting changes according to your defined policy.
* Reporting Service: Generates reports based on booking details, supplier information, financial data, and any other relevant data.
* Transport Management Service: Manages the details of transport suppliers, coordinates transport bookings and detects opportunities for combining transport bookings.
* Employee Management Service: Manages employee data, travel plans, and any other specific employee-related requirements.
* Feedback Service: Captures and saves feedback from employees about their bookings.

Each of these microservices will have its own area of responsibility and will interact with others as needed. With containerization and orchestration using GKE, is possible to run, manage, and scale these microservices efficiently.

### [SLAs](https://cloud.google.com/kubernetes-engine/sla)

|  |  |
| --- | --- |
| Covered Service | Monthly Uptime Percentage |
| Zonal Cluster (control plane) | 99.5% |
| Regional Cluster (control plane) | 99.95% |
| Autopilot Cluster (control plane) | 99.95% |
| Autopilot Pods in Multiple Zones | 99.9% |
| GKE Enterprise Autopilot Pods in Multiple Regions | 99.99% |

### GKE Pricing

GKE Enterprise offers pay-as-you-go pricing, where you are billed for GKE Enterprise managed clusters as you use them at the rates listed below. You can start using pay-as-you-go GKE Enterprise whenever you like by following the instructions in our [setup guides](https://cloud.google.com/anthos/docs/setup/overview).

Prices are listed in U.S. dollars (USD). If you pay in a currency other than USD, the prices listed in your currency on [Cloud Platform SKUs](https://cloud.google.com/skus) apply. A bill is sent out at the end of each billing cycle, listing previous usage and charges.

A screenshot of a computer

Description automatically generated

## Google Pubsub

Pub/Sub works as a messaging middleware for traditional service integration or a simple communication medium for modern microservices. Push subscriptions deliver events to serverless webhooks on Cloud Functions, App Engine, Cloud Run, or custom environments on Google Kubernetes Engine or Compute Engine.

### Why Pubsub?

* Scalability: Pub/Sub can automatically scale to meet application's needs, handling hundreds of millions of messages per second.
* Reliability: Pub/Sub guarantees at-least-once message delivery along with real-time consistency across data centers.
* Durability: Once a message is published, it's stored redundantly and won't be lost.
* Secure: All data is encrypted at rest and in transit.
* Global: Pub/Sub is designed as a global service; it is very effective if the application needs to communicate across different regions.
* Loose Coupling: The publisher and subscriber applications can evolve separately, and don't need to know details of each other. This helps in maintaining and updating the applications.
* Integrated: It's integrated with most of Google Cloud's data processing tools, making it easy to use for real-time analytics, Machine Learning, and other data integration pipelines.
* Versatile Delivery: It supports push delivery to HTTP/HTTPS endpoints and pull delivery, giving developers flexibility.
* Ordering: For those use-cases that need ordering, Pub/Sub ordering guarantees the delivery of messages in the order of their publishing.
* Dead-letter topics: Handling of failed messages is facilitated by dead-letter topics, which are used to sideline and inspect unprocessable messages.

### [SLAS](https://cloud.google.com/pubsub/sla)

|  |  |
| --- | --- |
| **Covered Service** | ****Monthly Uptime Percentage**** |
| Pub/Sub | >=99.95% |
| Pub/Sub Lite | >=99.95% for regional topics  >=99.5% for zonal topics |

### [Pubsub pricing](https://cloud.google.com/pubsub/pricing)

Every calendar month, the first 10 GiB of throughput identified as the Message Delivery Basic SKU for a billing account is free. After that, the price is $40 per TiB in all Google Cloud regions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Publish throughput in MiBps | Number of subscriptions | Zonal Lite topic | Regional Lite topic | Pub/Sub |
| 10 | 1 | $169 | $608 | $2,000 |
| 10 | 2 | $214 | $788 | $3,000 |
| 100 | 1 | $1,688 | $6,075 | $19,760 |
| 100 | 2 | $2,138 | $7,875 | $29,640 |

## Google Dataflow

Google Cloud Dataflow is a fully managed, scalable data processing service for executing batch, stream, and ETL processing patterns.

### Why Dataflow?

* Fully Managed: You don't have to worry about underlying infrastructure management. Dataflow will automatically handle task distribution and resource management for you.
* Scalability: Dataflow can automatically scale up and down depending on the job requirement. It can handle large amount of data with efficiency.
* Real-Time and Batch Processing: Dataflow supports both batch and real-time data processing. You can choose to stream changes in Firestore to BigQuery in real-time or do it in batch mode.
* Fault-Tolerant: Dataflow ensures reliable and consistent processing by providing exactly once processing of data.
* Flexibility: You can use various transformation options and even apply custom transformations to process the data as you transfer it.
* Integration: Dataflow easily integrates with other GCP services. In this case, using Firestore and BigQuery with Dataflow would be straight-forward.
* Uniform Programming Model: Dataflow uses Apache Beam SDK which provides a uniform programming model that can handle both batch and stream data.
* Monitoring and Debugging: It provides monitoring through Stackdriver. Visual representation of jobs using Dataflow UI simplifies debugging and understanding of pipeline execution.

### [SLAs](https://cloud.google.com/dataflow/sla)

|  |  |
| --- | --- |
| **Covered Service** | **Monthly Uptime Percentage** |
| Dataflow | >=99.9% |

### Dataflow pricing

A screenshot of a computer

Description automatically generated

## Google BigQuery

BigQuery is a fully managed enterprise data warehouse that helps you manage and analyze your data with built-in features like machine learning, geospatial analysis, and business intelligence.

### Why BigQuery

Google BigQuery is a highly scalable, serverless, fully managed, and highly durable multi-cloud data warehouse designed for business agility. It comes with a built-in machine learning capability. Here's why BigQuery is a good match for your scenario:

* Fully Managed: BigQuery requires no infrastructure to manage, allowing you to focus on analyzing data rather than managing infrastructure.
* Highly Scalable: BigQuery can handle the scale from gigabytes to petabytes of data and provide high-speed analysis. It can automatically scale up or down based on the data load.
* Real-time Analytics: BigQuery supports real-time analytics through its high-speed streaming insertion API.
* SQL Queries: BigQuery uses SQL, making the analysis straightforward for all users who know SQL. You can also use User Defined Functions (UDFs) in SQL queries.
* Integration: It is well integrated with other Google Cloud services and can be connected to many reporting and data visualization tools such as Google Data Studio, Tableau, and others.
* ML Integration: BigQuery ML enables data scientists and data analysts to build and operationalize ML models on structured data directly inside BigQuery using SQL.
* Geospatial data: BigQuery GIS lets you analyze and visualize geospatial data, which can be useful if you want to use location-based data.
* Storage: You can store huge amount of data very cheaply for long-term analysis.
* Security: Google BigQuery follows strong security measures like encryption at rest and in transit, VPC Service Controls, and dedicated IAM roles.

It's an ideal tool for running analytics over large datasets, providing you insights in real-time. That's why it is considered an industry-leading and powerful data warehouse technology.

### [SLAs](https://cloud.google.com/bigquery/sla)

|  |  |
| --- | --- |
| Covered Service | Monthly Uptime Percentage |
| BigQuery (except BigQuery Standard edition) | >= 99.99% |
| BigQuery (Standard edition) | >= 99.9% |

Additionally, BigQuery Data Transfer Service will include a Data Delivery Service Level Objective as follows:

|  |  |
| --- | --- |
| Covered Service | Data Delivery |
| BigQuery Data Transfer Service | Less than or equal to 24 hours |

### [BigQuery pricing](https://cloud.google.com/bigquery/pricing)

A screenshot of a web page

Description automatically generated

# Availability

According to Google's SLA, here are the monthly availability percentages for each service:

Google Kubernetes Engine (GKE)\*\*: 99.5%

Cloud Firestore: 99.999% (multi-region), 99.99% (Regional)

Cloud Pub/Sub: 99.95%

Cloud Dataflow: 99.9%

BigQuery: 99.9%

## Operational Service

|  |  |  |
| --- | --- | --- |
| **Service** | **Individual SLA** | **Calculation Value** |
| GKE | 99.5% | 0.995 |
| Firestore | 99.999% | 0.99999 |
| Pub/Sub | 99.95% | 0.9995 |
| **Composite SLA (Operational Pipeline)** | - | **0.99449 (99.449%)** |

Composite SLA = 99.5/100 \* 99.999/100 \* 99.95/100 = 0.995 \* 0.99999 \* 0.9995 = 0.99449

## Analytics Service

|  |  |  |
| --- | --- | --- |
| **Service** | **Individual SLA** | **Calculation Value** |
| Firestore | 99.999% | 0.99999 |
| Dataflow | 99.9% | 0.999 |
| BigQuery | 99.9% | 0.999 |
| **Composite SLA (Analytical Pipeline)** | - | **0.99889 (99.889%)** |

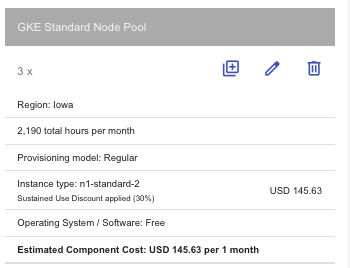
Composite SLA = 99.999/100 \* 99.9/100 \* 99.9/100 = 0.99999 \* 0.999 \* 0.999 = 0.99889

To minimize service disruption, you should also consider Google’s recommended best practices around reliability, like setting up health checks, design for redundancy, implementing retrial mechanisms, etc.

# Cost calculation

Key considerations for each component:

Google Kubernetes Engine (GKE): Costs depend on the cluster size and type, as well as its uptime. Also, consider the persistent disk costs and network egress costs.



Cloud Firestore: Costs depend on the number of reads, writes, and deletes, as well as the amount of data stored.

A screenshot of a computer

Description automatically generated

Cloud Pub/Sub: Costs depend on the volume of messages and the size of each message.

A screenshot of a phone

Description automatically generated

Cloud Dataflow: Costs depend on the resources (such as vCPU, memory, and storage) used by the Dataflow job, and the job execution time.

A screenshot of a computer

Description automatically generated

BigQuery: Costs depend on the amount of data stored, the complexity and amount of queries processed, and the frequency of data ingestion.

A screenshot of a computer

Description automatically generated