Cloud-Native AI SaaS

Multimodal Object Detection

5CCSACCA Coursework Phase 4

https://github.com/5CCSACCA/coursework-Minigo-ovo.git

Kexin Wang

Student ID: k23168350

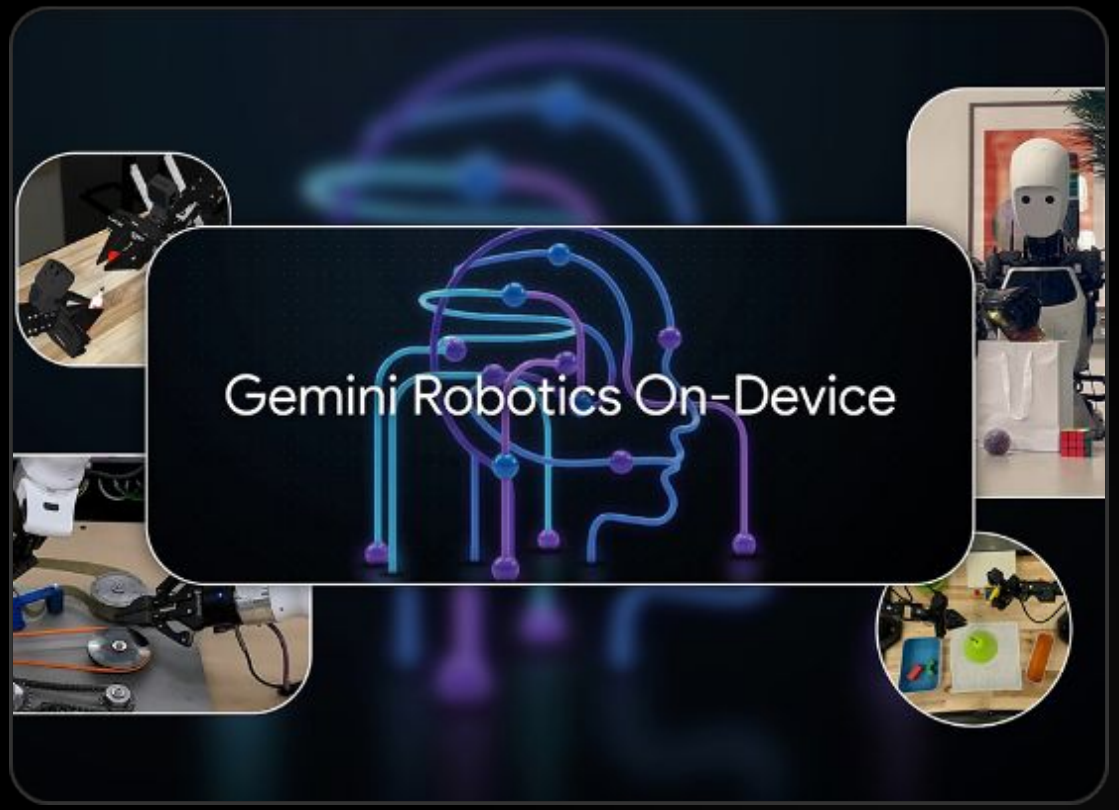
System Overview

Goal

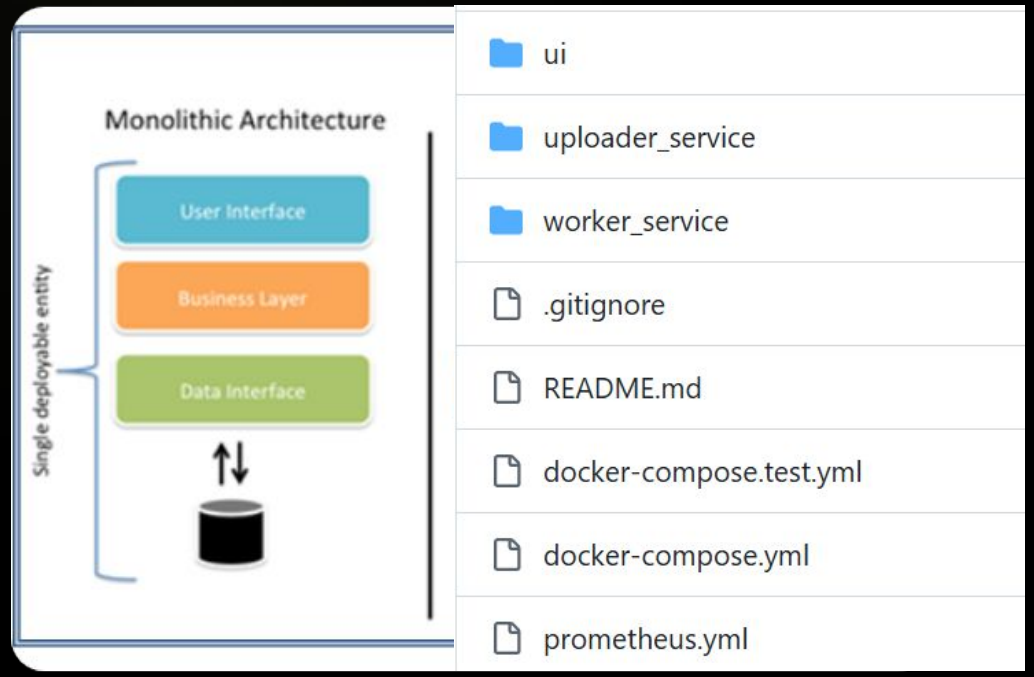
Build a scalable, cloud-native SaaS for advanced image and text processing.

Core Capabilities

- **Multimodal Input:** Handles both public Image URLs and Natural Language prompts.
- **Output:** Detailed AI-generated descriptions and reasoning.
- **Key Features:** Asynchronous Processing, RESTful API, and Real-time Monitoring.



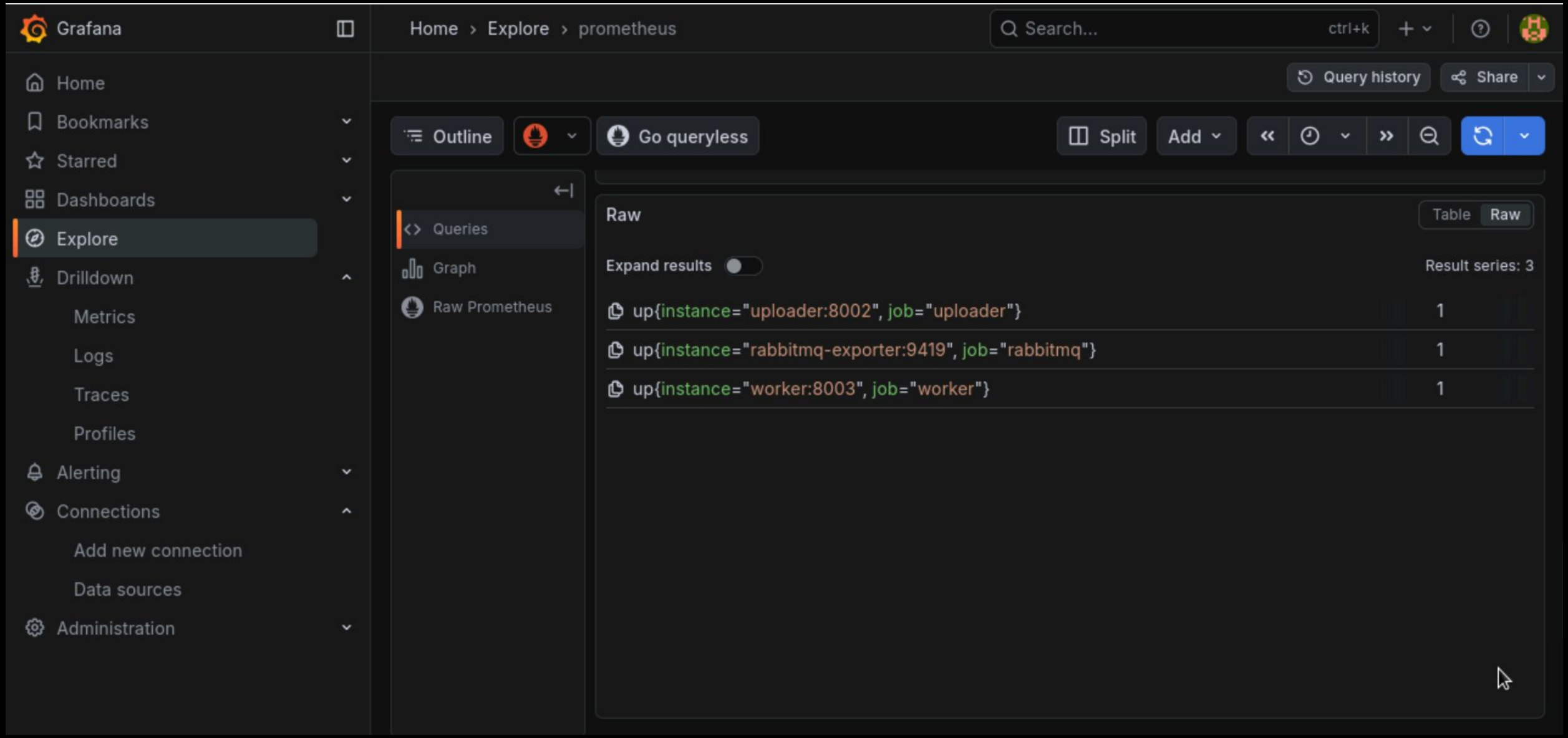
System Architecture

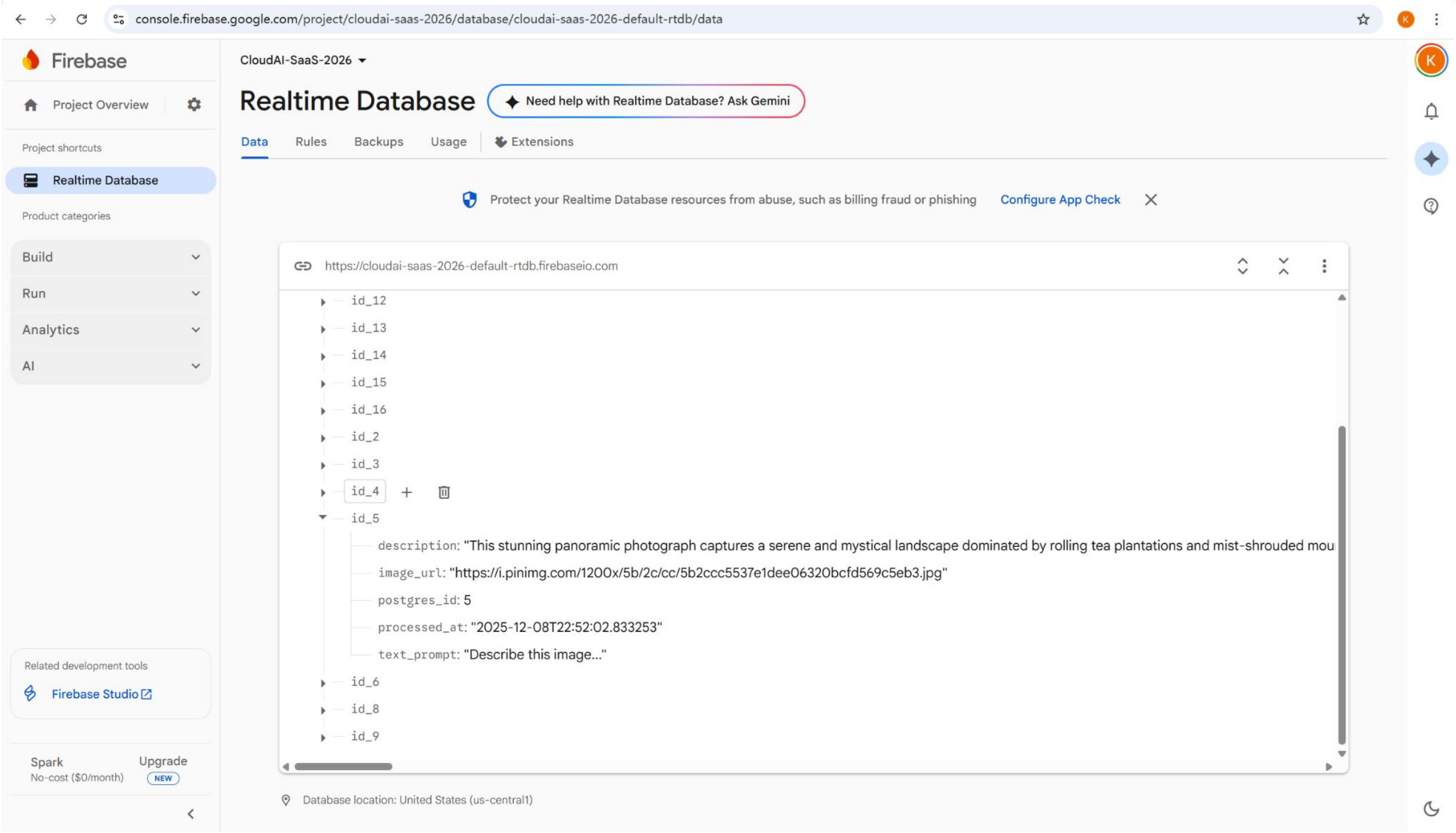


- ui
- uploader_service
- worker_service
- .gitignore
- README.md
- docker-compose.test.yml
- docker-compose.yml
- prometheus.yml

Producer-Consumer Pattern

- **Uploader (API Gateway):** FastAPI service that accepts user requests and queues them.
- **RabbitMQ:** Message broker ensuring decoupling and load buffering.
- **Worker:** Background service that processes tasks and calls AI models.
- **Storage:** PostgreSQL for metadata; Firebase for unstructured results.


Grafana
Home
Bookmarks
Starred
Dashboards
Explore
Drilldown
Metrics
Logs
Traces
Profiles
Alerting
Connections
Add new connection
Data sources
Administration
Home › Explore › prometheus
Search... ctrl+k
Query history
Share
Outline
Go queryless
Split
Add
Queries
Graph
Raw Prometheus
Raw
Table
Raw
Expand results
Result series: 3
up{instance="uploader:8002", job="uploader"} 1
up{instance="rabbitmq-exporter:9419", job="rabbitmq"} 1
up{instance="worker:8003", job="worker"} 1


console.firebase.google.com/project/cloudai-saas-2026/database/cloudai-saas-2026-default-rtdb/data
Firebase
CloudAI-SaaS-2026
Project Overview
Project shortcuts
Realtime Database
Product categories
Build
Run
Analytics
AI
Related development tools
Firebase Studio
Spark
No-cost ($0/month)
Upgrade
NEW
Realtime Database
Need help with Realtime Database? Ask Gemini
Data
Rules
Backups
Usage
Extensions
Protect your Realtime Database resources from abuse, such as billing fraud or phishing
Configure App Check
https://cloudai-saas-2026-default-rtdb.firebaseio.com
id_12
id_13
id_14
id_15
id_16
id_2
id_3
id_4
id_5
description: "This stunning panoramic photograph captures a serene and mystical landscape dominated by rolling tea plantations and mist-shrouded mou
image_url: "https://i.pinimg.com/1200x/5b/2c/cc/5b2ccc5537e1dee06320bcfd569c5eb3.jpg"
postgres_id: 5
processed_at: "2025-12-08T22:52:02.833253"
text_prompt: "Describe this image..."
id_6
id_8
id_9
Database location: United States (us-central1)

AI Model Strategy

Selected Model

Google Gemini 2.5 Flash

A high-performance multimodal model capable of understanding both visual and textual inputs simultaneously.

Strategic Advantages

- **Multimodal:** Replaces the need for separate YOLO (vision) and LLM (text) pipelines.
- **Efficiency:** Offloads heavy inference computation to the cloud, reducing local resource usage.
- **Integration:** Seamlessly integrated via the `google-genai` SDK within the Worker service.

Development & CI/CD

GitFlow Workflow

Development followed a strict branching strategy: Main for release, Develop for integration, and Feature/* for isolated tasks.

Containerization

Fully containerized using **Docker** to ensure consistency across development and production environments.

Automation

Deployment is simplified to a single command: docker compose up.

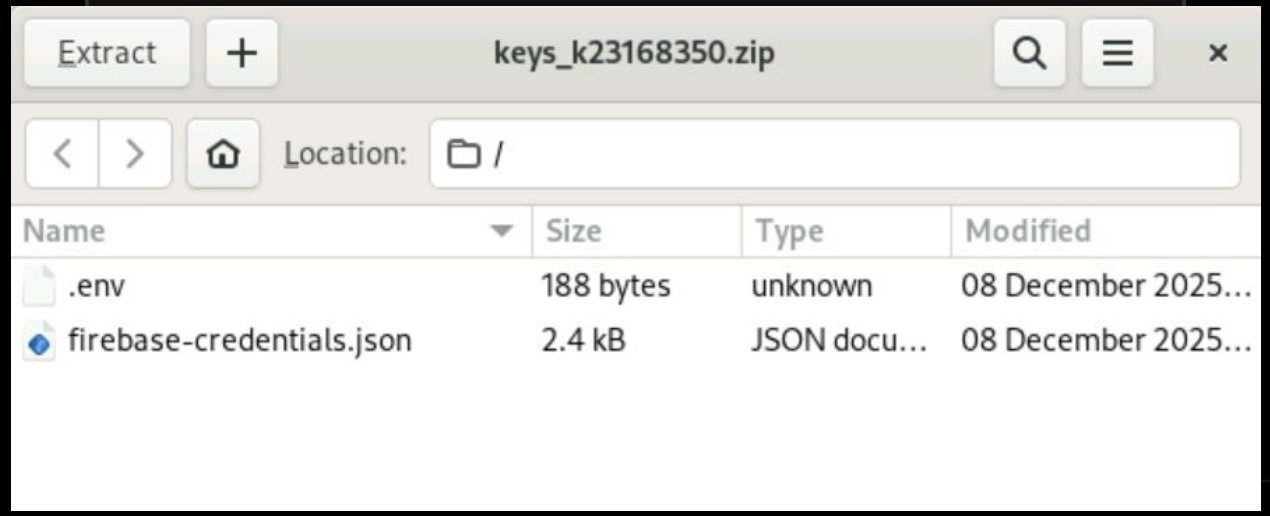
Quality Assurance

docker-compose.test.yml

systemtest.bash

Testing command in one line

```
sudo docker compose -f
docker-compose.test.yml up --build
--abort-on-container-exit
```



Testing

Implemented an automated integration test suite (`unittest`) covering the full pipeline from submission to result retrieval.

Security

Credentials managed via `.env` files (never hardcoded). Database ports are isolated from the public network.

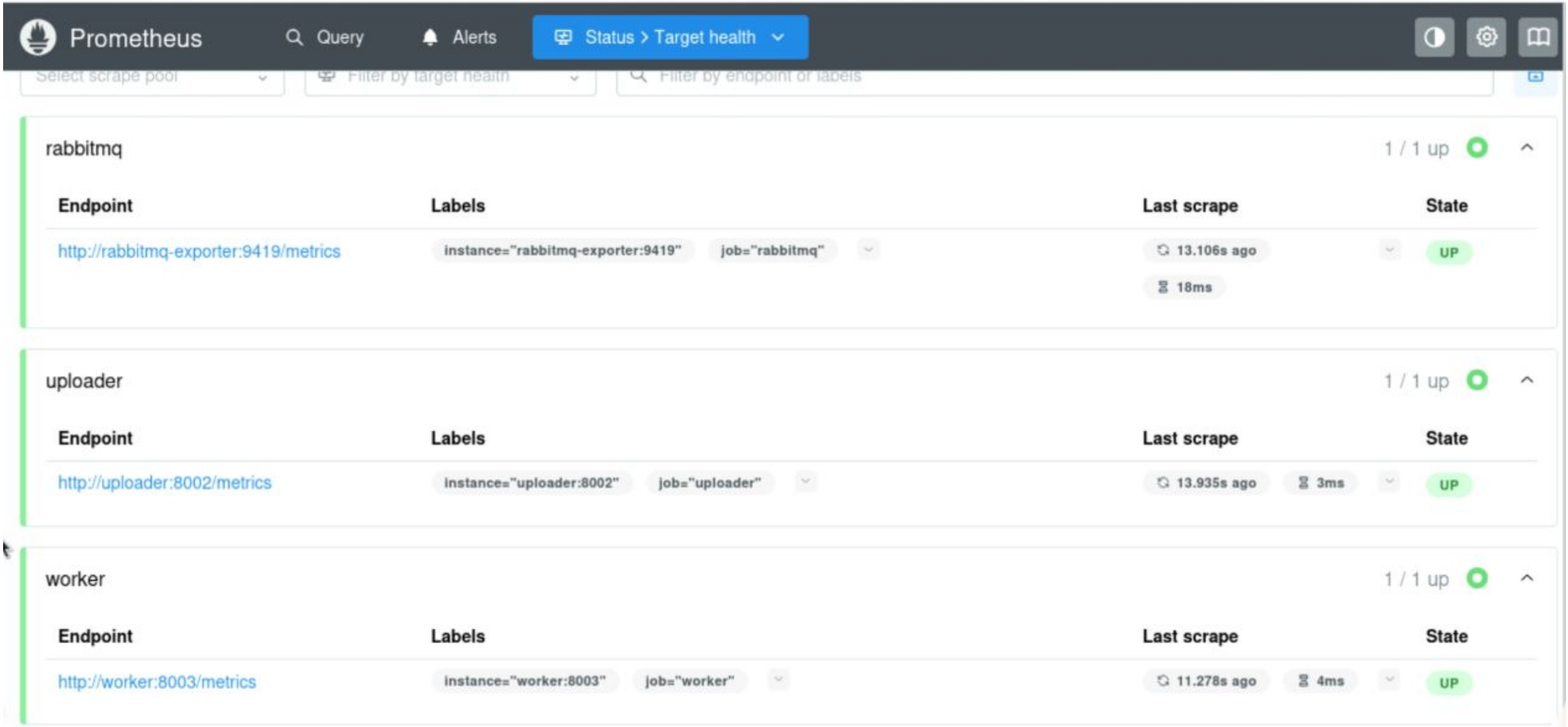
Monitoring

Prometheus collects metrics (traffic, latency), and **Grafana** visualizes system health in real-time.

```
system-tests-1  | [Test] Running system tests...
system-tests-1  | ============================= test session starts ==============================
system-tests-1  | platform linux -- Python 3.10.19, pytest-9.0.2, pluggy-1.6.0 -- /usr/local/bin/python3.10
system-tests-1  | cachedir: .pytest_cache
system-tests-1  | rootdir: /app
system-tests-1  | plugins: anyio-4.12.0
system-tests-1  | collecting ... collected 1 item
system-tests-1  |
uploader-1      | Uploader: Prometheus metrics server started on port 8002
uploader-1      | Uploader: Database initialized.
uploader-1      | Uploader: Firebase initialized.
uploader-1      | INFO:     172.18.0.6:56070 - "GET /health HTTP/1.1" 200 OK
rabbitmq-1      | 2025-12-09 03:42:01.229155+00:00 [info] <0.847.0> accepting AMQP connection <0.847.0> (172.18.0.4:54778 -> 172.18.0.3:5672
)
rabbitmq-1      | 2025-12-09 03:42:01.232729+00:00 [info] <0.847.0> connection <0.847.0> (172.18.0.4:54778 -> 172.18.0.3:5672): user 'guest'
 authenticated and granted access to vhost '/'
uploader-1      | INFO:     172.18.0.6:56080 - "POST /submit_task HTTP/1.1" 200 OK
rabbitmq-1      | 2025-12-09 03:42:01.257413+00:00 [warning] <0.847.0> closing AMQP connection <0.847.0> (172.18.0.4:54778 -> 172.18.0.3:567
2, vhost: '/', user: 'guest'):
rabbitmq-1      | 2025-12-09 03:42:01.257413+00:00 [warning] <0.847.0> client unexpectedly closed TCP connection
worker-1        | /app/worker_consumer.py:38: MovedIn20Warning: The ``declarative_base()`` function is now available as sqlalchemy.orm.decla
rative_base(). (deprecated since: 2.0) (Background on SQLAlchemy 2.0 at: https://sqlalche.me/e/b8d9)
worker-1        |   Base = declarative_base()
rabbitmq-1      | 2025-12-09 03:42:02.313164+00:00 [info] <0.870.0> accepting AMQP connection <0.870.0> (172.18.0.5:33260 -> 172.18.0.3:5672
)
rabbitmq-1      | 2025-12-09 03:42:02.317301+00:00 [info] <0.870.0> connection <0.870.0> (172.18.0.5:33260 -> 172.18.0.3:5672): user 'guest'
 authenticated and granted access to vhost '/'
system-tests-1  | systemtest.py::TestSystemIntegration::test_full_async_flow
 Container my_submission-system-tests-1  Stopped
 Container my_submission-worker-1  Stopping
```

```
rabbitmq-1      | 2025-12-09 03:42:14.187442+00:00 [notice] <0.64.0>
rabbitmq-1      | 2025-12-09 03:42:14.189864+00:00 [warning] <0.704.0> HTTP listener registry could not find context rabbitmq_prometheus_tls
db-1            | 2025-12-09 03:42:14.195 UTC [1] LOG:  received fast shutdown request
db-1            | 2025-12-09 03:42:14.199 UTC [1] LOG:  aborting any active transactions
db-1            | 2025-12-09 03:42:14.206 UTC [1] LOG:  background worker "logical replication launcher" (PID 60) exited with exit code 1
db-1            | 2025-12-09 03:42:14.212 UTC [55] LOG:  shutting down
rabbitmq-1      | 2025-12-09 03:42:14.206692+00:00 [warning] <0.704.0> HTTP listener registry could not find context rabbitmq_management_tls
db-1            | 2025-12-09 03:42:14.216 UTC [55] LOG:  checkpoint starting: shutdown immediate
rabbitmq-1      | 2025-12-09 03:42:14.221377+00:00 [info] <0.840.0> stopped TCP listener on [::]:5672
rabbitmq-1      | 2025-12-09 03:42:14.226285+00:00 [info] <0.632.0> Virtual host '/' is stopping
rabbitmq-1      | 2025-12-09 03:42:14.226508+00:00 [info] <0.889.0> Closing all connections in vhost '/' on node 'rabbit@88d4e5b2240b' becau
se the vhost is stopping
rabbitmq-1      | 2025-12-09 03:42:14.238264+00:00 [info] <0.645.0> Stopping message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88
d4e5b2240b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_persistent'
rabbitmq-1      | 2025-12-09 03:42:14.249408+00:00 [info] <0.645.0> Message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88d4e5b2240
b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_persistent' is stopped
rabbitmq-1      | 2025-12-09 03:42:14.249877+00:00 [info] <0.641.0> Stopping message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88
d4e5b2240b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_transient'
db-1            | 2025-12-09 03:42:14.255 UTC [55] LOG:  checkpoint complete: wrote 60 buffers (0.4%); 0 WAL file(s) added, 0 removed, 0 rec
ycled; write=0.013 s, sync=0.006 s, total=0.043 s; sync files=45, longest=0.004 s, average=0.001 s; distance=170 kB, estimate=170 kB
rabbitmq-1      | 2025-12-09 03:42:14.270534+00:00 [info] <0.641.0> Message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88d4e5b2240
b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_transient' is stopped
db-1            | 2025-12-09 03:42:14.275 UTC [1] LOG:  database system is shut down
 Container my_submission-db-1  Stopped
db-1 exited with code 0
 Container my_submission-rabbitmq-1  Stopped
rabbitmq-1 exited with code 0

k23168350@cloud-computing-for-ai-2526-v2-l111-u2659896:~/my_submission$
w Enable Watch
```


http://localhost:9090/targets
Prometheus
Query
Alerts
Status > Target health
rabbitmq
1 / 1 up
Endpoint
Labels
Last scrape
State
http://rabbitmq-exporter:9419/metrics
instance="rabbitmq-exporter:9419"
job="rabbitmq"
13.106s ago
18ms
UP
uploader
1 / 1 up
Endpoint
Labels
Last scrape
State
http://uploader:8002/metrics
instance="uploader:8002"
job="uploader"
13.935s ago
3ms
UP
worker
1 / 1 up
Endpoint
Labels
Last scrape
State
http://worker:8003/metrics
instance="worker:8003"
job="worker"
11.278s ago
4ms
UP


Graph
Lines
Bars
Points
Stacked lines
Stacked bars
1
0.8
0.6
0.4
0.2
0
08:55
09:00
09:05
09:10
09:15
09:20
09:25
09:30
09:35
09:40
09:45
09:50
{__name__="up", instance="rabbitmq-exporter:9419", job="rabbitmq"}
{__name__="up", instance="uploader:8002", job="uploader"}
{__name__="up", instance="worker:8003", job="worker"}

Optimization: RabbitMQ buffers traffic spikes, preventing the need to provision peak capacity instantly.

Cost Estimation

Unit Metrics

- **Backend VM (AWS c6gd.large):**
  - **Capacity: 100 users per VM**
  - **Cost (C_b): $35.40 / month (derived from $354 for 10 VMs)**
- **Frontend VM (AWS c6gd.large):**
  - **Capacity: 10,000 users per VM**
  - **Cost (C_f): $35.40 / month**

Calculation

Scenario A: Scaling to 1,000 Users

- **Backend: $1000 \text{ users} / 100 = 10 \text{ VMs} \times \$35.4 = \$354.0$**
- **Frontend: $1000 \text{ users} / 10000 = 1 \text{ VM} \times \$35.4 = \$35.4$**
- **Total Monthly Cost: $389.40**

Scenario B: Scaling to 500 Users (Minimum Price Calculation)

- **Backend: $500 \text{ users} / 100 = 5 \text{ VMs} \times \$35.4 = \$177.0$**
- **Frontend: $1 \text{ VM} \text{ (sufficient for 10k)} \times \$35.4 = \$35.4$**
- **Total Monthly Cost: $177.0 + 35.4 = \mathbf{\$212.40}$**
- **Cost Per User: $\$212.40 / 500 = \mathbf{\$0.42}$**

The Formula

$$TotalCost = \lceil \frac{TU}{U_b} \rceil \times C_b + \lceil \frac{TU}{U_f} \rceil \times C_f + F$$

Analysis

Fixed Costs: The API Gateway is lightweight and stateless.

Variable Costs: The **Worker** is the most expensive component due to processing intensity. To handle 100k users, we auto-scale the number of Worker instances (N).

Sustainability & Limitations

Sustainability

The event-driven asynchronous architecture allows resources (Workers) to idle or scale down when the queue is empty, significantly reducing energy consumption.

Limitations

- **Input:** Currently relies on publicly accessible image URLs; no local file upload support yet.
- **Latency:** Dependent on external API network speeds and rate limits.

References

Documentation

- FastAPI: fastapi.tiangolo.com
- Docker: docs.docker.com
- Google Gemini API: aistudio.google.com
- Firebase: firebase.google.com/docs

Tools

- RabbitMQ: rabbitmq.com
- Prometheus & Grafana: prometheus.io, grafana.com
- Streamlit: streamlit.io

Image Sources

https://www.therobotreport.com/wp-content/uploads/2025/06/gemini-featured.jpg

Source: www.therobotreport.com

https://blog.sysfore.com/wp-content/uploads/2016/01/Microservices-Architecture.png

Source: blog.sysfore.com

https://learn.microsoft.com/en-us/azure/devops/pipelines/architectures/media/azure-devops-ci-cd-architecture.svg?view=azure-devops

Source: learn.microsoft.com

https://pub.mdpi-res.com/sensors/sensors-25-00079/article_deploy/html/images/sensors-25-00079-ag.png?1735281637

Source: www.mdpi.com

https://www.accrets.com/wp-content/uploads/Green-Cloud-Computing-A-sustainable-way-01.png

Source: www.accrets.com

Live Demonstration

1 Minute System

Walkthrough

Cloud-Native AI SaaS

Multimodal Object Detection

5CCSACCA Coursework Phase 4

https://github.com/5CCSACCA/coursework-Minigo-ovo.git

Kexin Wang

Student ID: k23168350

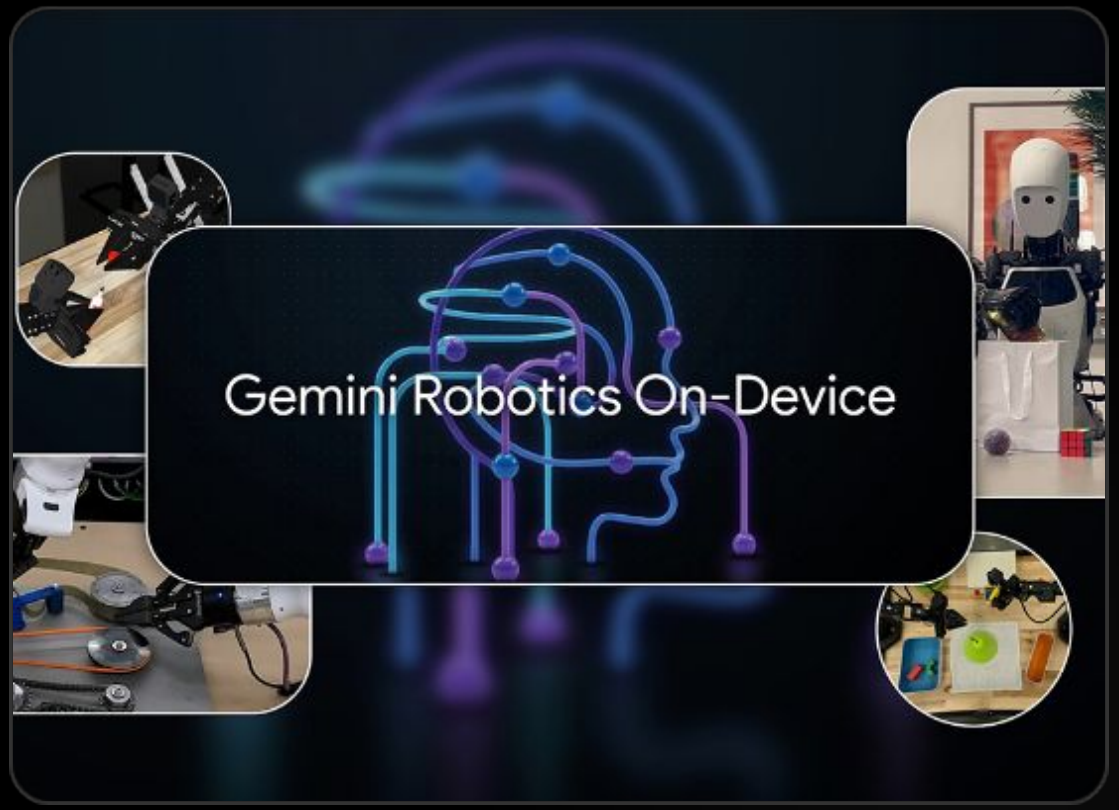
System Overview

Goal

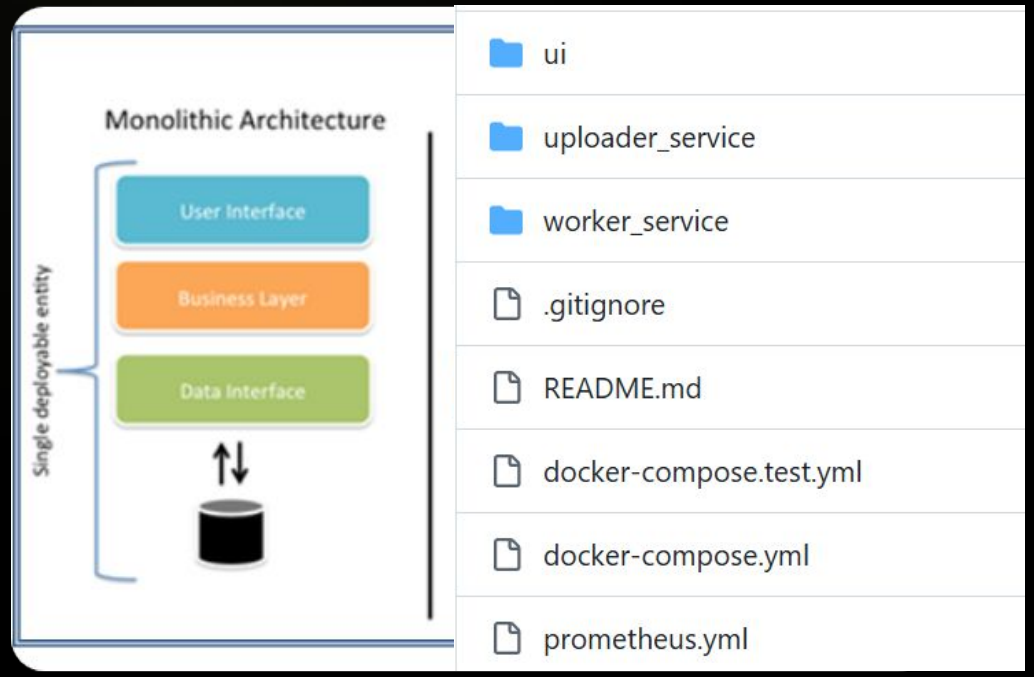
Build a scalable, cloud-native SaaS for advanced image and text processing.

Core Capabilities

- **Multimodal Input:** Handles both public Image URLs and Natural Language prompts.
- **Output:** Detailed AI-generated descriptions and reasoning.
- **Key Features:** Asynchronous Processing, RESTful API, and Real-time Monitoring.



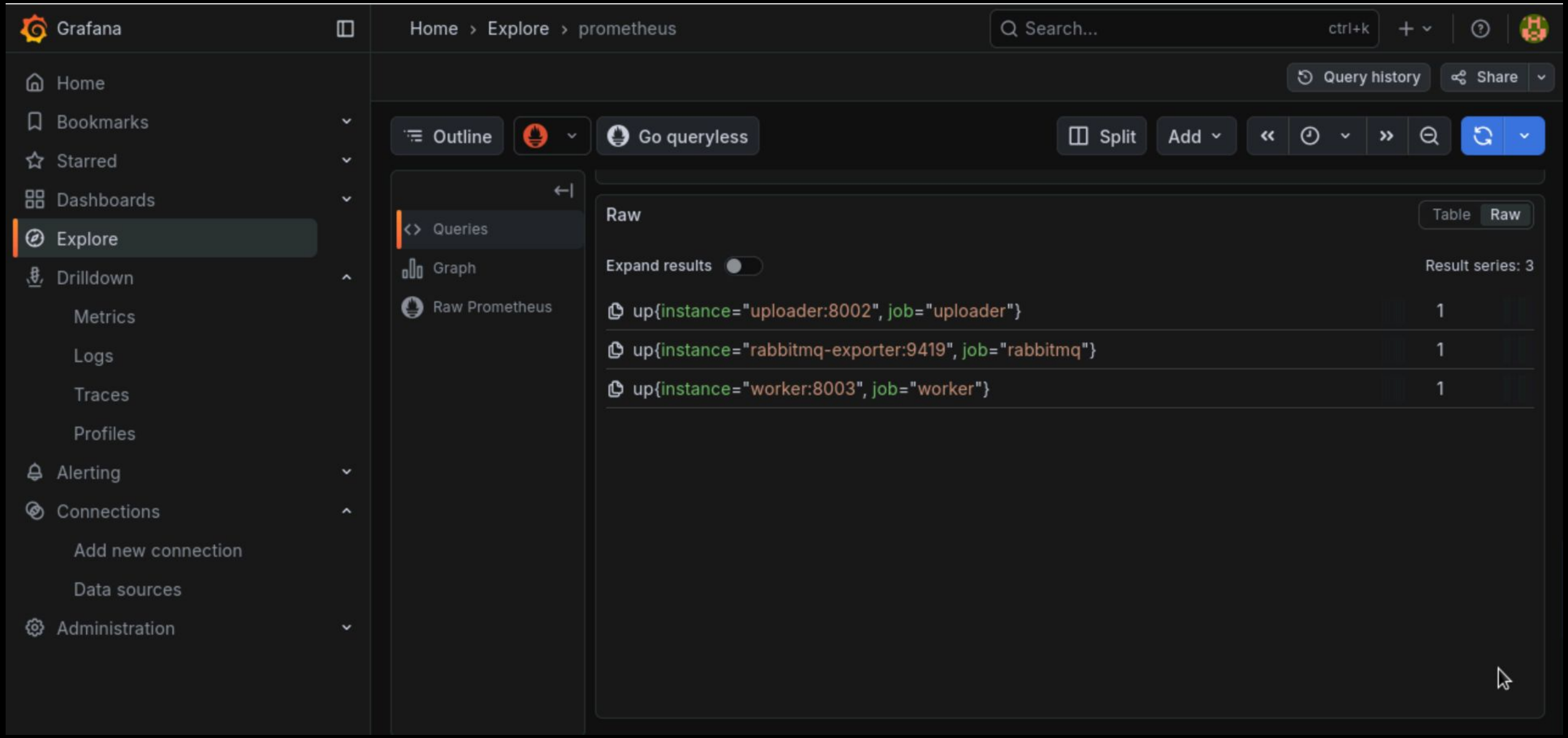
System Architecture

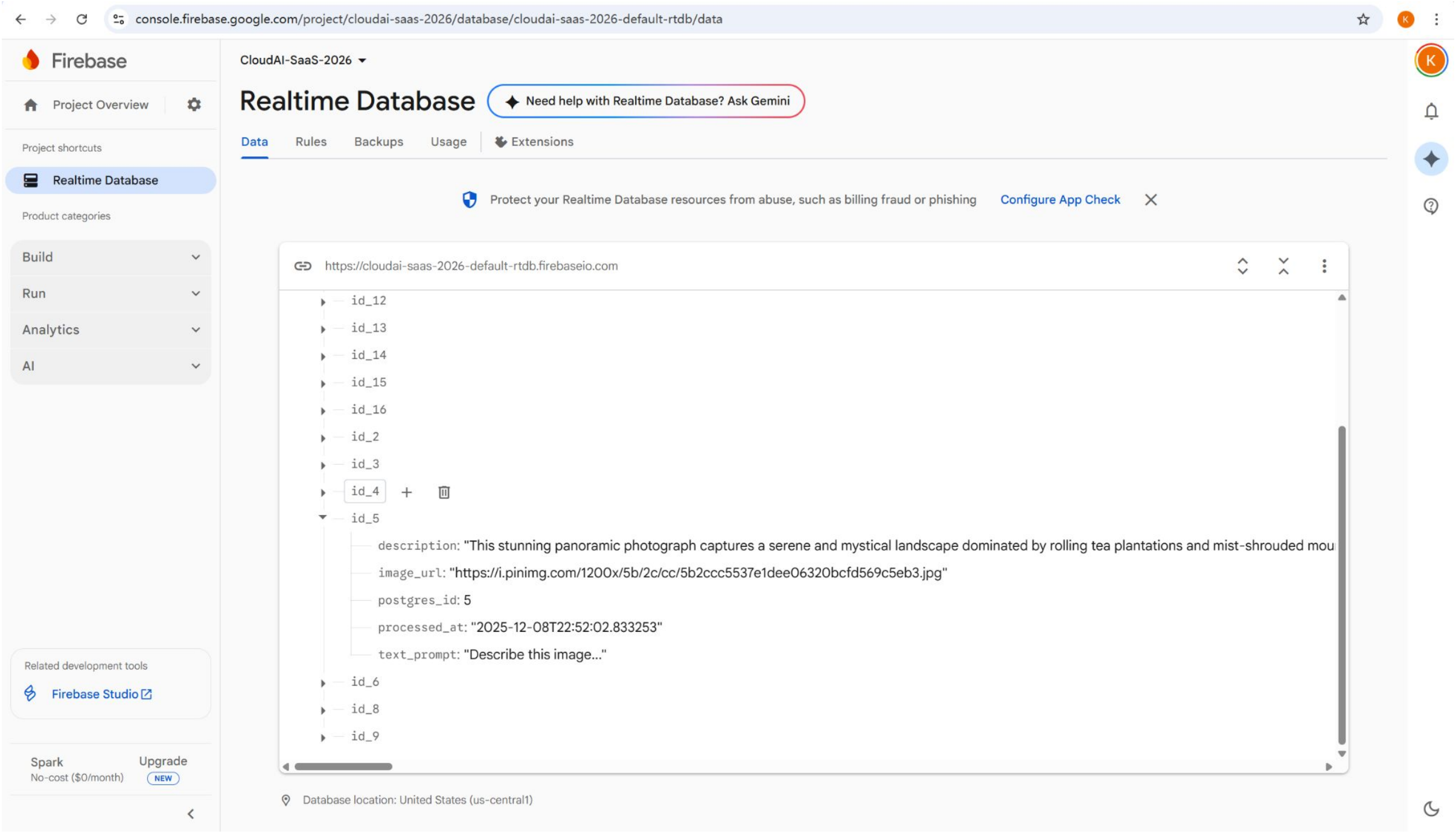


- ui
- uploader_service
- worker_service
- .gitignore
- README.md
- docker-compose.test.yml
- docker-compose.yml
- prometheus.yml

Producer-Consumer Pattern

- **Uploader (API Gateway):** FastAPI service that accepts user requests and queues them.
- **RabbitMQ:** Message broker ensuring decoupling and load buffering.
- **Worker:** Background service that processes tasks and calls AI models.
- **Storage:** PostgreSQL for metadata; Firebase for unstructured results.


Grafana
Home
Bookmarks
Starred
Dashboards
Explore
Drilldown
Metrics
Logs
Traces
Profiles
Alerting
Connections
Add new connection
Data sources
Administration
Home › Explore › prometheus
Search... ctrl+k
Query history
Share
Outline
Go queryless
Split
Add
Queries
Graph
Raw Prometheus
Raw
Table
Raw
Expand results
Result series: 3
up{instance="uploader:8002", job="uploader"} 1
up{instance="rabbitmq-exporter:9419", job="rabbitmq"} 1
up{instance="worker:8003", job="worker"} 1


console.firebase.google.com/project/cloudai-saas-2026/database/cloudai-saas-2026-default-rtdb/data
Firebase
CloudAI-SaaS-2026
Project Overview
Project shortcuts
Realtime Database
Product categories
Build
Run
Analytics
AI
Related development tools
Firebase Studio
Spark
No-cost ($0/month)
Upgrade
NEW
Realtime Database
Need help with Realtime Database? Ask Gemini
Data
Rules
Backups
Usage
Extensions
Protect your Realtime Database resources from abuse, such as billing fraud or phishing
Configure App Check
https://cloudai-saas-2026-default-rtdb.firebaseio.com
id_12
id_13
id_14
id_15
id_16
id_2
id_3
id_4
id_5
description: "This stunning panoramic photograph captures a serene and mystical landscape dominated by rolling tea plantations and mist-shrouded mou
image_url: "https://i.pinimg.com/1200x/5b/2c/cc/5b2ccc5537e1dee06320bcfd569c5eb3.jpg"
postgres_id: 5
processed_at: "2025-12-08T22:52:02.833253"
text_prompt: "Describe this image..."
id_6
id_8
id_9
Database location: United States (us-central1)

AI Model Strategy

Selected Model

Google Gemini 2.5 Flash

A high-performance multimodal model capable of understanding both visual and textual inputs simultaneously.

Strategic Advantages

- **Multimodal:** Replaces the need for separate YOLO (vision) and LLM (text) pipelines.
- **Efficiency:** Offloads heavy inference computation to the cloud, reducing local resource usage.
- **Integration:** Seamlessly integrated via the `google-genai` SDK within the Worker service.

Development & CI/CD

GitFlow Workflow

Development followed a strict branching strategy: Main for release, Develop for integration, and Feature/* for isolated tasks.

Containerization

Fully containerized using **Docker** to ensure consistency across development and production environments.

Automation

Deployment is simplified to a single command: docker compose up.

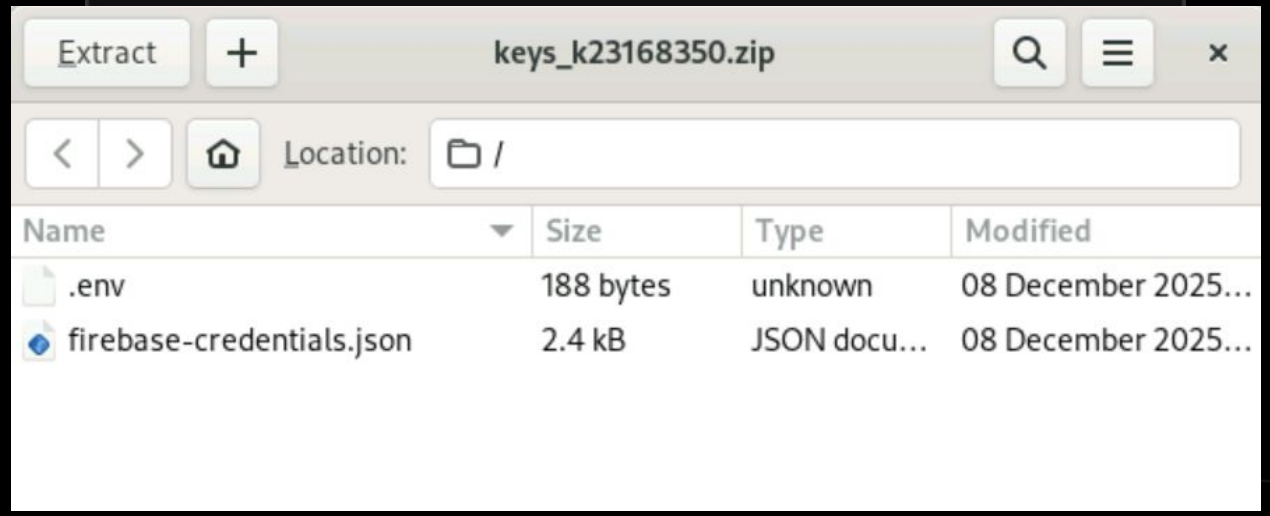
Quality Assurance

docker-compose.test.yml

systemtest.bash

Testing command in one line

```
sudo docker compose -f
docker-compose.test.yml up --build
--abort-on-container-exit
```



Testing

Implemented an automated integration test suite (`unittest`) covering the full pipeline from submission to result retrieval.

Security

Credentials managed via `.env` files (never hardcoded). Database ports are isolated from the public network.

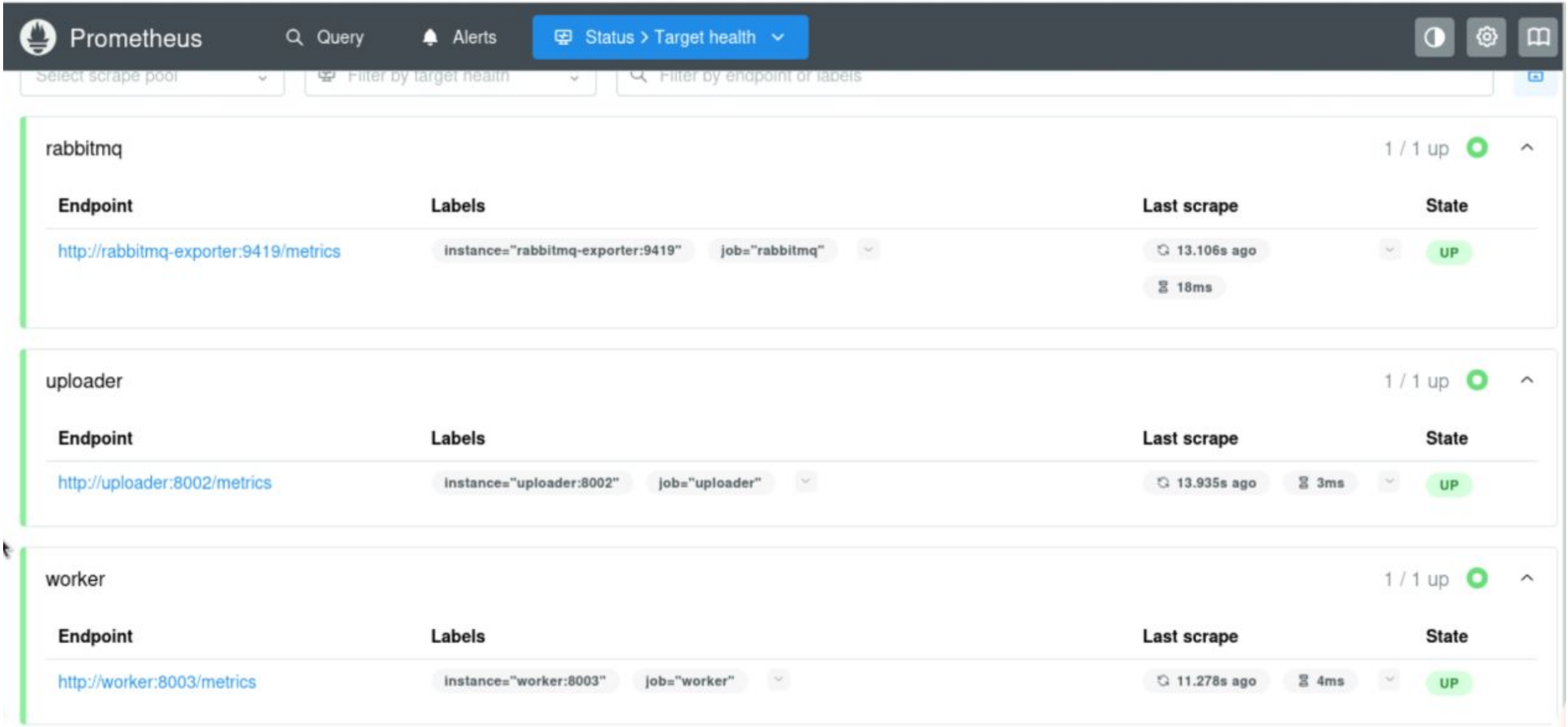
Monitoring

Prometheus collects metrics (traffic, latency), and **Grafana** visualizes system health in real-time.

```
system-tests-1  | [Test] Running system tests...
system-tests-1  | ============================= test session starts ==============================
system-tests-1  | platform linux -- Python 3.10.19, pytest-9.0.2, pluggy-1.6.0 -- /usr/local/bin/python3.10
system-tests-1  | cachedir: .pytest_cache
system-tests-1  | rootdir: /app
system-tests-1  | plugins: anyio-4.12.0
system-tests-1  | collecting ... collected 1 item
system-tests-1  |
uploader-1      | Uploader: Prometheus metrics server started on port 8002
uploader-1      | Uploader: Database initialized.
uploader-1      | Uploader: Firebase initialized.
uploader-1      | INFO:     172.18.0.6:56070 - "GET /health HTTP/1.1" 200 OK
rabbitmq-1      | 2025-12-09 03:42:01.229155+00:00 [info] <0.847.0> accepting AMQP connection <0.847.0> (172.18.0.4:54778 -> 172.18.0.3:5672
)
rabbitmq-1      | 2025-12-09 03:42:01.232729+00:00 [info] <0.847.0> connection <0.847.0> (172.18.0.4:54778 -> 172.18.0.3:5672): user 'guest'
 authenticated and granted access to vhost '/'
uploader-1      | INFO:     172.18.0.6:56080 - "POST /submit_task HTTP/1.1" 200 OK
rabbitmq-1      | 2025-12-09 03:42:01.257413+00:00 [warning] <0.847.0> closing AMQP connection <0.847.0> (172.18.0.4:54778 -> 172.18.0.3:567
2, vhost: '/', user: 'guest'):
rabbitmq-1      | 2025-12-09 03:42:01.257413+00:00 [warning] <0.847.0> client unexpectedly closed TCP connection
worker-1        | /app/worker_consumer.py:38: MovedIn20Warning: The ``declarative_base()`` function is now available as sqlalchemy.orm.decla
rative_base(). (deprecated since: 2.0) (Background on SQLAlchemy 2.0 at: https://sqlalche.me/e/b8d9)
worker-1        |   Base = declarative_base()
rabbitmq-1      | 2025-12-09 03:42:02.313164+00:00 [info] <0.870.0> accepting AMQP connection <0.870.0> (172.18.0.5:33260 -> 172.18.0.3:5672
)
rabbitmq-1      | 2025-12-09 03:42:02.317301+00:00 [info] <0.870.0> connection <0.870.0> (172.18.0.5:33260 -> 172.18.0.3:5672): user 'guest'
 authenticated and granted access to vhost '/'
system-tests-1  | systemtest.py::TestSystemIntegration::test_full_async_flow
 Container my_submission-system-tests-1  Stopped
 Container my_submission-worker-1  Stopping
```

```
rabbitmq-1      | 2025-12-09 03:42:14.187442+00:00 [notice] <0.64.0>
rabbitmq-1      | 2025-12-09 03:42:14.189864+00:00 [warning] <0.704.0> HTTP listener registry could not find context rabbitmq_prometheus_tls
db-1            | 2025-12-09 03:42:14.195 UTC [1] LOG:  received fast shutdown request
db-1            | 2025-12-09 03:42:14.199 UTC [1] LOG:  aborting any active transactions
db-1            | 2025-12-09 03:42:14.206 UTC [1] LOG:  background worker "logical replication launcher" (PID 60) exited with exit code 1
db-1            | 2025-12-09 03:42:14.212 UTC [55] LOG:  shutting down
rabbitmq-1      | 2025-12-09 03:42:14.206692+00:00 [warning] <0.704.0> HTTP listener registry could not find context rabbitmq_management_tls
db-1            | 2025-12-09 03:42:14.216 UTC [55] LOG:  checkpoint starting: shutdown immediate
rabbitmq-1      | 2025-12-09 03:42:14.221377+00:00 [info] <0.840.0> stopped TCP listener on [::]:5672
rabbitmq-1      | 2025-12-09 03:42:14.226285+00:00 [info] <0.632.0> Virtual host '/' is stopping
rabbitmq-1      | 2025-12-09 03:42:14.226508+00:00 [info] <0.889.0> Closing all connections in vhost '/' on node 'rabbit@88d4e5b2240b' becau
se the vhost is stopping
rabbitmq-1      | 2025-12-09 03:42:14.238264+00:00 [info] <0.645.0> Stopping message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88
d4e5b2240b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_persistent'
rabbitmq-1      | 2025-12-09 03:42:14.249408+00:00 [info] <0.645.0> Message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88d4e5b2240
b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_persistent' is stopped
rabbitmq-1      | 2025-12-09 03:42:14.249877+00:00 [info] <0.641.0> Stopping message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88
d4e5b2240b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_transient'
db-1            | 2025-12-09 03:42:14.255 UTC [55] LOG:  checkpoint complete: wrote 60 buffers (0.4%); 0 WAL file(s) added, 0 removed, 0 rec
ycled; write=0.013 s, sync=0.006 s, total=0.043 s; sync files=45, longest=0.004 s, average=0.001 s; distance=170 kB, estimate=170 kB
rabbitmq-1      | 2025-12-09 03:42:14.270534+00:00 [info] <0.641.0> Message store for directory '/var/lib/rabbitmq/mnesia/rabbit@88d4e5b2240
b/msg_stores/vhosts/628WB79CIFDYO9LJI6DKMI09L/msg_store_transient' is stopped
db-1            | 2025-12-09 03:42:14.275 UTC [1] LOG:  database system is shut down
 Container my_submission-db-1  Stopped
db-1 exited with code 0
 Container my_submission-rabbitmq-1  Stopped
rabbitmq-1 exited with code 0

k23168350@cloud-computing-for-ai-2526-v2-l111-u2659896:~/my_submission$
w Enable Watch
```


http://localhost:9090/targets
Prometheus
Query
Alerts
Status > Target health
rabbitmq
1 / 1 up
Endpoint
Labels
Last scrape
State
http://rabbitmq-exporter:9419/metrics
instance="rabbitmq-exporter:9419"
job="rabbitmq"
13.106s ago
18ms
UP
uploader
1 / 1 up
Endpoint
Labels
Last scrape
State
http://uploader:8002/metrics
instance="uploader:8002"
job="uploader"
13.935s ago
3ms
UP
worker
1 / 1 up
Endpoint
Labels
Last scrape
State
http://worker:8003/metrics
instance="worker:8003"
job="worker"
11.278s ago
4ms
UP


Graph
Lines
Bars
Points
Stacked lines
Stacked bars
1
0.8
0.6
0.4
0.2
0
08:55
09:00
09:05
09:10
09:15
09:20
09:25
09:30
09:35
09:40
09:45
09:50
{__name__="up", instance="rabbitmq-exporter:9419", job="rabbitmq"}
{__name__="up", instance="uploader:8002", job="uploader"}
{__name__="up", instance="worker:8003", job="worker"}

Optimization: RabbitMQ buffers traffic spikes, preventing the need to provision peak capacity instantly.

Cost Estimation

Unit Metrics

- **Backend VM (AWS c6gd.large):**
  - **Capacity: 100 users per VM**
  - **Cost (C_b): $35.40 / month (derived from $354 for 10 VMs)**
- **Frontend VM (AWS c6gd.large):**
  - **Capacity: 10,000 users per VM**
  - **Cost (C_f): $35.40 / month**

Calculation

Scenario A: Scaling to 1,000 Users

- **Backend: $1000 \text{ users} / 100 = 10 \text{ VMs} \times \$35.4 = \$354.0$**
- **Frontend: $1000 \text{ users} / 10000 = 1 \text{ VM} \times \$35.4 = \$35.4$**
- **Total Monthly Cost: $389.40**

Scenario B: Scaling to 500 Users (Minimum Price Calculation)

- **Backend: $500 \text{ users} / 100 = 5 \text{ VMs} \times \$35.4 = \$177.0$**
- **Frontend: $1 \text{ VM} \text{ (sufficient for 10k)} \times \$35.4 = \$35.4$**
- **Total Monthly Cost: $177.0 + 35.4 = \mathbf{\$212.40}$**
- **Cost Per User: $\$212.40 / 500 = \mathbf{\$0.42}$**

The Formula

$$TotalCost = \lceil \frac{TU}{U_b} \rceil \times C_b + \lceil \frac{TU}{U_f} \rceil \times C_f + F$$

Analysis

Fixed Costs: The API Gateway is lightweight and stateless.

Variable Costs: The **Worker** is the most expensive component due to processing intensity. To handle 100k users, we auto-scale the number of Worker instances (N).

Sustainability & Limitations

Sustainability

The event-driven asynchronous architecture allows resources (Workers) to idle or scale down when the queue is empty, significantly reducing energy consumption.

Limitations

- **Input:** Currently relies on publicly accessible image URLs; no local file upload support yet.
- **Latency:** Dependent on external API network speeds and rate limits.

References

Documentation

- FastAPI: fastapi.tiangolo.com
- Docker: docs.docker.com
- Google Gemini API: aistudio.google.com
- Firebase: firebase.google.com/docs

Tools

- RabbitMQ: rabbitmq.com
- Prometheus & Grafana: prometheus.io, grafana.com
- Streamlit: streamlit.io

Image Sources

https://www.therobotreport.com/wp-content/uploads/2025/06/gemini-featured.jpg

Source: www.therobotreport.com

https://blog.sysfore.com/wp-content/uploads/2016/01/Microservices-Architecture.png

Source: blog.sysfore.com

https://learn.microsoft.com/en-us/azure/devops/pipelines/architectures/media/azure-devops-ci-cd-architecture.svg?view=azure-devops

Source: learn.microsoft.com

https://pub.mdpi-res.com/sensors/sensors-25-00079/article_deploy/html/images/sensors-25-00079-ag.png?1735281637

Source: www.mdpi.com

https://www.accrets.com/wp-content/uploads/Green-Cloud-Computing-A-sustainable-way-01.png

Source: www.accrets.com

Live Demonstration

1 Minute System

Walkthrough