

# Pub/Sub – Plan efficient System Design

Setting up and Best Practices



Praveen Thirumurugan

Senior Software Engineer, Plum.  
He/Him/His



@praveentcom

**“Async systems are good.  
But shortcuts are not.”**

System design is an art

## Agenda

01

# What's Pub/Sub?

Firstly, let's deep dive on what's Pub/Sub and why is it used by small to large businesses.

02

# Setting up Pub/Sub

Did you know that it takes less than 10 minutes to make your legacy processes faster?

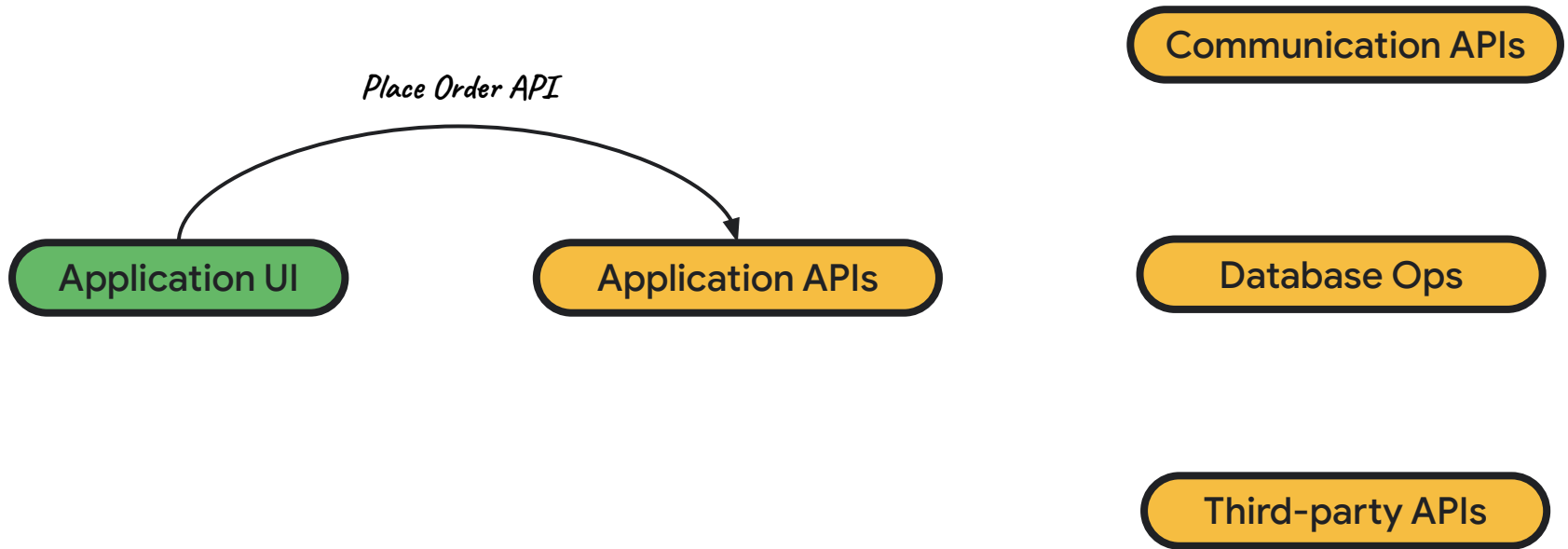
03

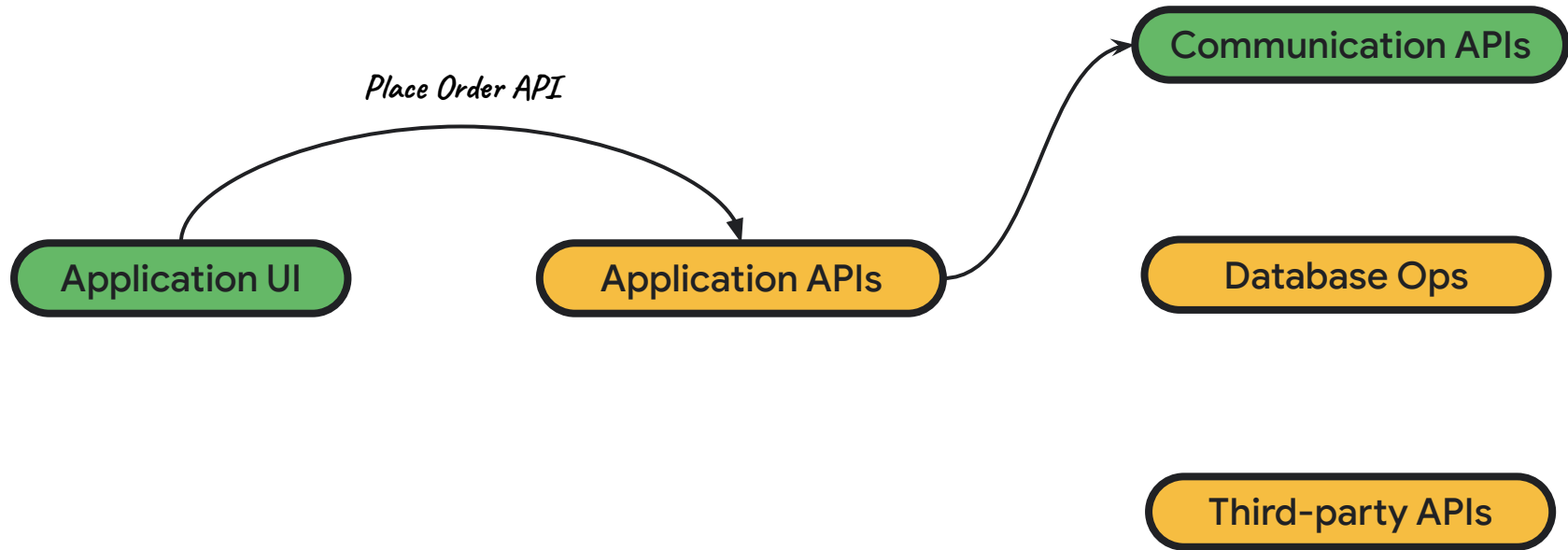
# Tl;dr Best Practices

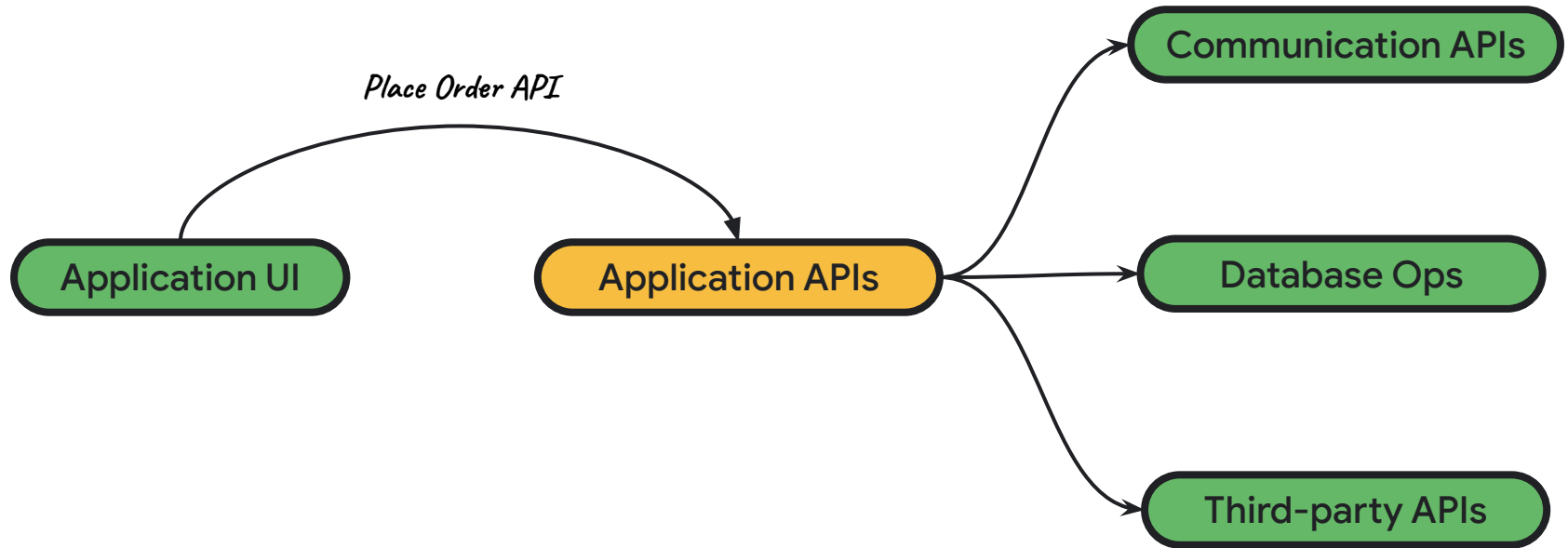
It's not the end, you need to keep your systems off from intruders & keep costs in check.

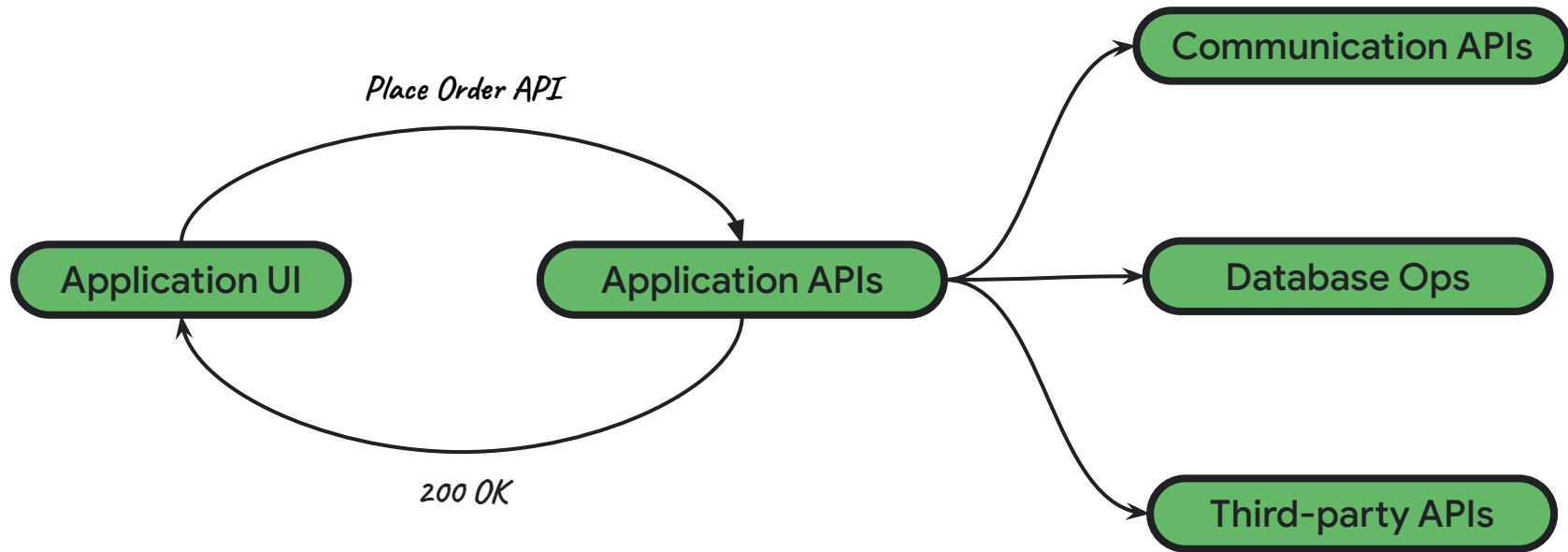
# What's Pub/Sub?

Firstly, let's deep dive on what's Pub/Sub and why is it used by small to large businesses.



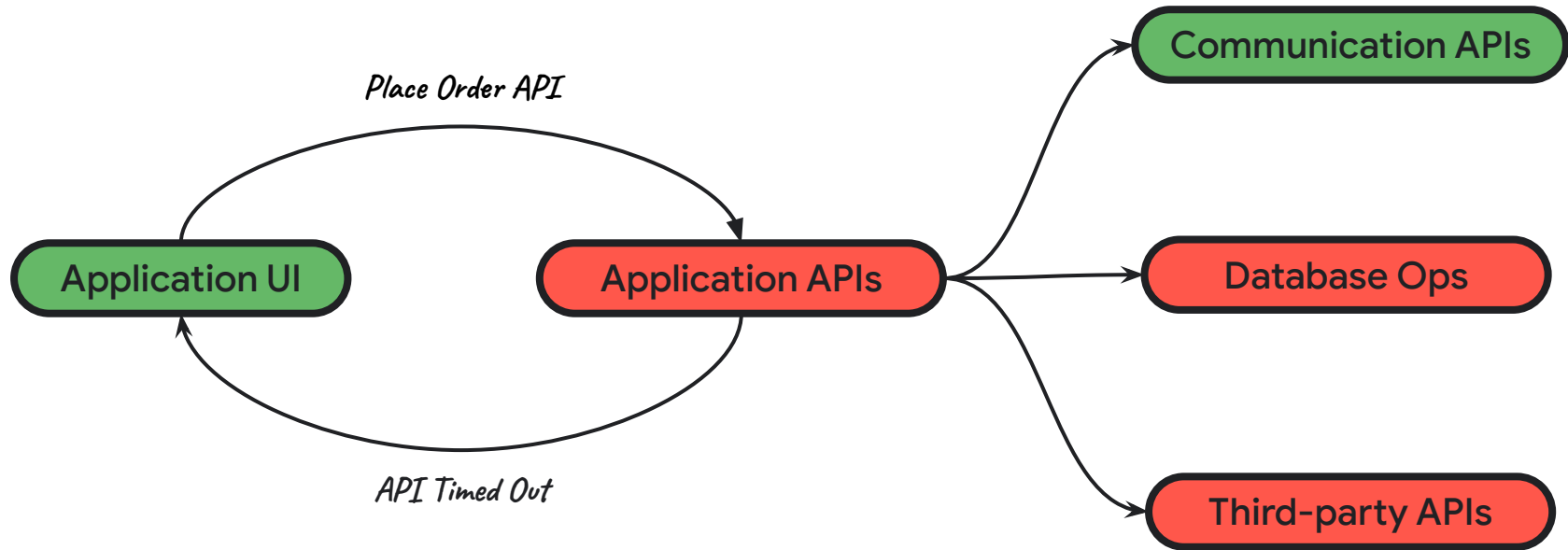




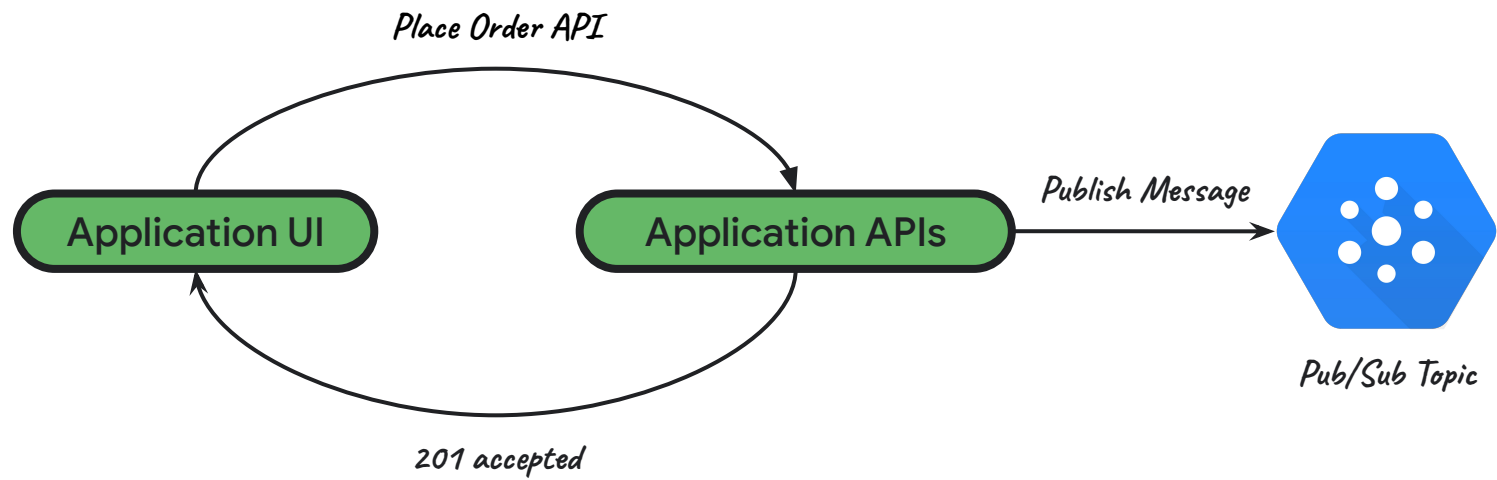


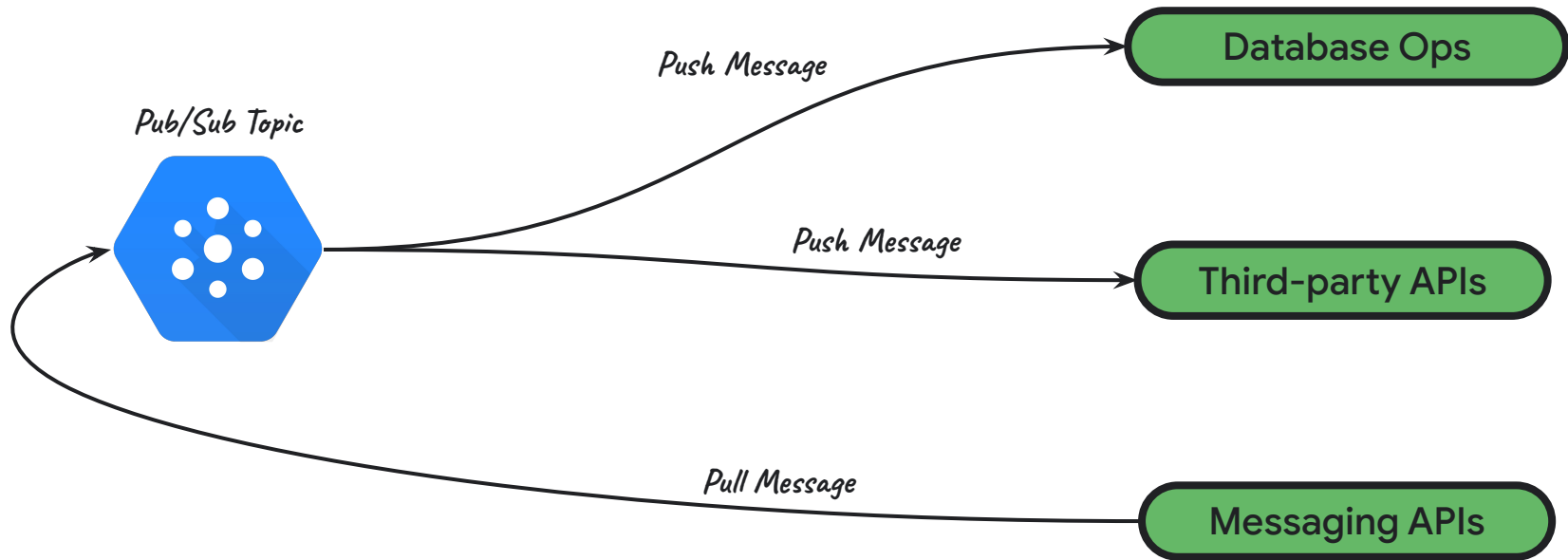


What's **wrong** with this approach?



How do you **fix** this  
problem then?





# Setting up Pub/Sub?

Did you know that it takes less than 10 minutes to make your legacy processes faster?

Google Cloud

Flock

SearchProducts, resources, docs (/)

Pub/Sub

Topics

CREATE TOPICDELETE

Topics

Subscriptions

Snapshots

Schemas

Lite reservations

Lite topics

Lite Subscriptions

Filter

Filter topics

	Topic ID ↑	Encryption key	Topic name	Retention		
<input checked="" type="checkbox"/>		Google-managed		—	⋮	▼
<input type="checkbox"/>		Google-managed		—	⋮	▼
<input type="checkbox"/>		Google-managed		—	⋮	▼
<input type="checkbox"/>		Google-managed		—	⋮	▼
<input type="checkbox"/>		Google-managed		—	⋮	▼
<input type="checkbox"/>		Google-managed		—	⋮	▼
<input type="checkbox"/>		Google-managed		—	⋮	▼

dead-letter-queue

PERMISSIONSLABELS

Edit or delete permissions below or 'Add principal' to grant new

Show inherited permissions

Filter

Enter property name or value

Role/Principal ↑

Cloud Scheduler Service Agent (1)

Editor (3)

Owner (1)

Pub/Sub Admin (1)

Pub/Sub Publisher (1)

<https://console.cloud.google.com/cloudpubsub>

Pub/Sub is a product offered by Google Cloud Platform and can be accessed via this URL once you have created a project.

For most small to medium workloads, and learning, your free tier is enough.

```
import { PubSub } from "@google-cloud/pubsub";

const pubsub = new PubSub({
  projectId: JSON.parse(process.env.SERVICE_ACCOUNT).project_id,
  credentials: {
    client_email: JSON.parse(process.env.SERVICE_ACCOUNT).client_email,
    private_key: JSON.parse(process.env.SERVICE_ACCOUNT).private_key,
  },
});

await pubsub.topic('topic').publishMessage({
  attributes: {
    "whereAmI": "Chennai"
  },
  data: Buffer.from('Hello from Cloud Community Days', 'utf8'),
});
```



# Tl;dr Best Practices

It's not the end, you need to keep your systems off from intruders & keep costs in check.

# Learn about your endpoint

In case your APIs are not responding fast or you think it takes too long to respond, this can be the place to set the timeout.

## Acknowledgement deadline ?

Seconds

Deadline time is from 10 seconds to 600 seconds

# Enable Dead Lettering

This ensures that you are not pushing in infinite loop in case things go wrong.

## Dead lettering

☒ Enable dead lettering

Subscriptions may configure a maximum number of delivery attempts. When cannot be delivered, it is republished to the specified dead-letter topic.

Select a dead-letter topic \*

Maximum delivery attempts \*

5

Maximum delivery attempts is from 5 to 100.

# Enable Authentication

JWT Authentication  
with Google Service Account

## Delivery type

☐ Pull

☒ Push

Endpoint URL \*

☒ Enable authentication [Learn more](#)

Service account \*

Audience (optional)

```
const jwtToken = req.headers.get("authorization)?.substring(7);

const claim = jose.decodeJwt(jwtToken);

if (
  !claim.email_verified ||
  claim.aud !== `https://example.com` ||
  claim.email !== JSON.parse(process.env.SERVICE_ACCOUNT).client_email ||
  claim.azp !== JSON.parse(process.env.SERVICE_ACCOUNT).client_id
) {
  // The request is not authenticated by Google
}

// aud = Audience Intended
// azp = Authorised Recipient
// email = Service Account Email
```

# Thank you!



Praveen Thirumurugan

Senior Software Engineer, Plum.  
He/Him/His



@praveentcom