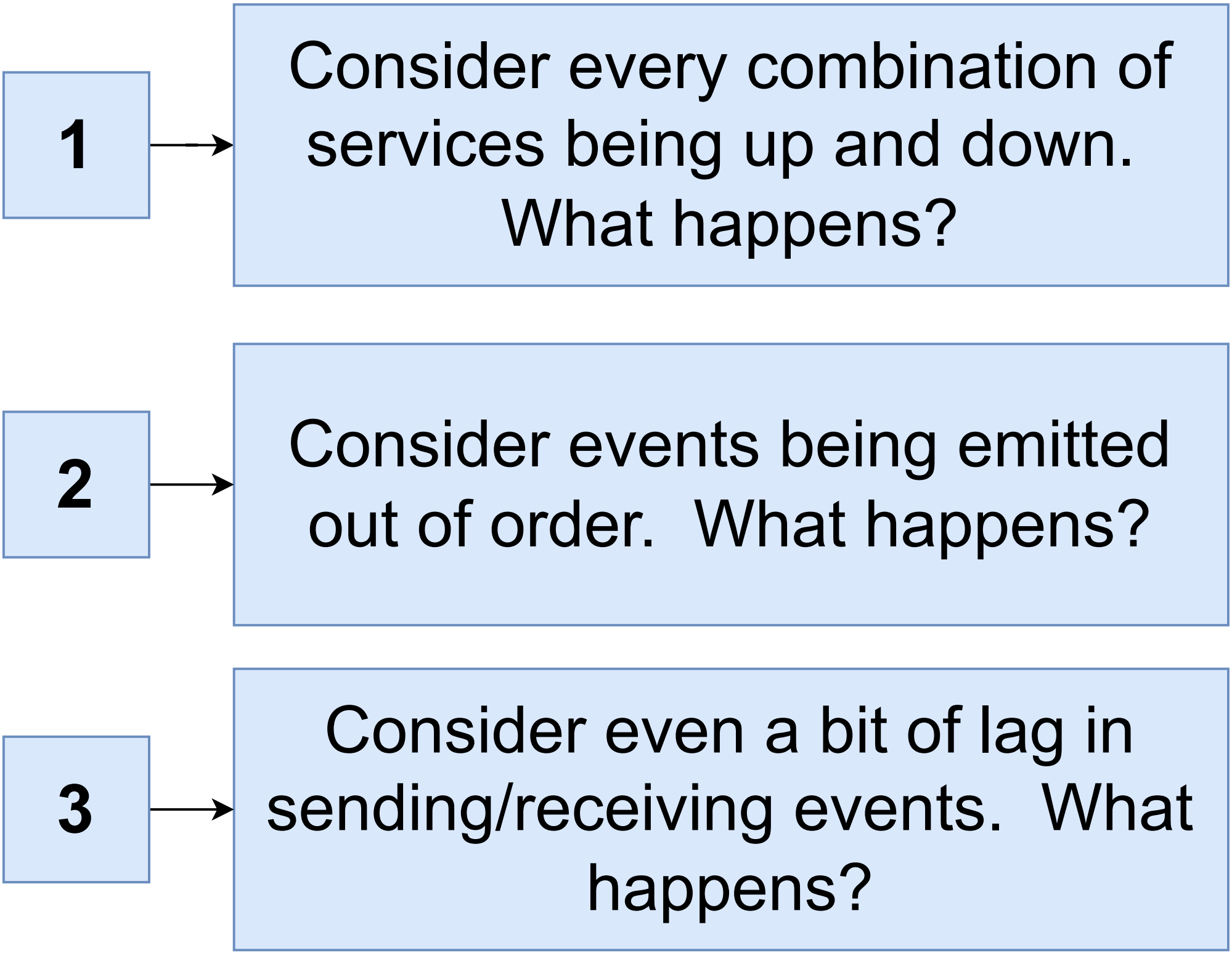
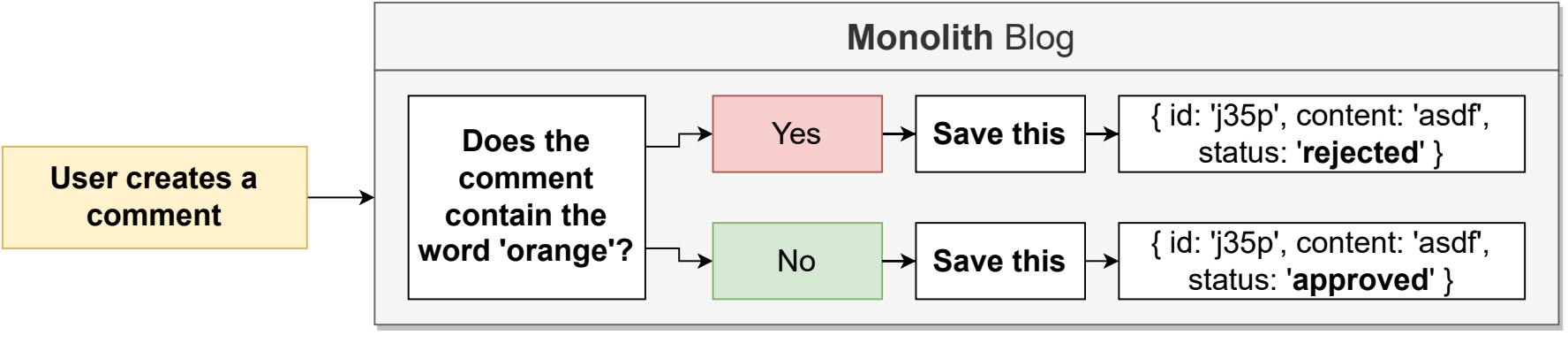
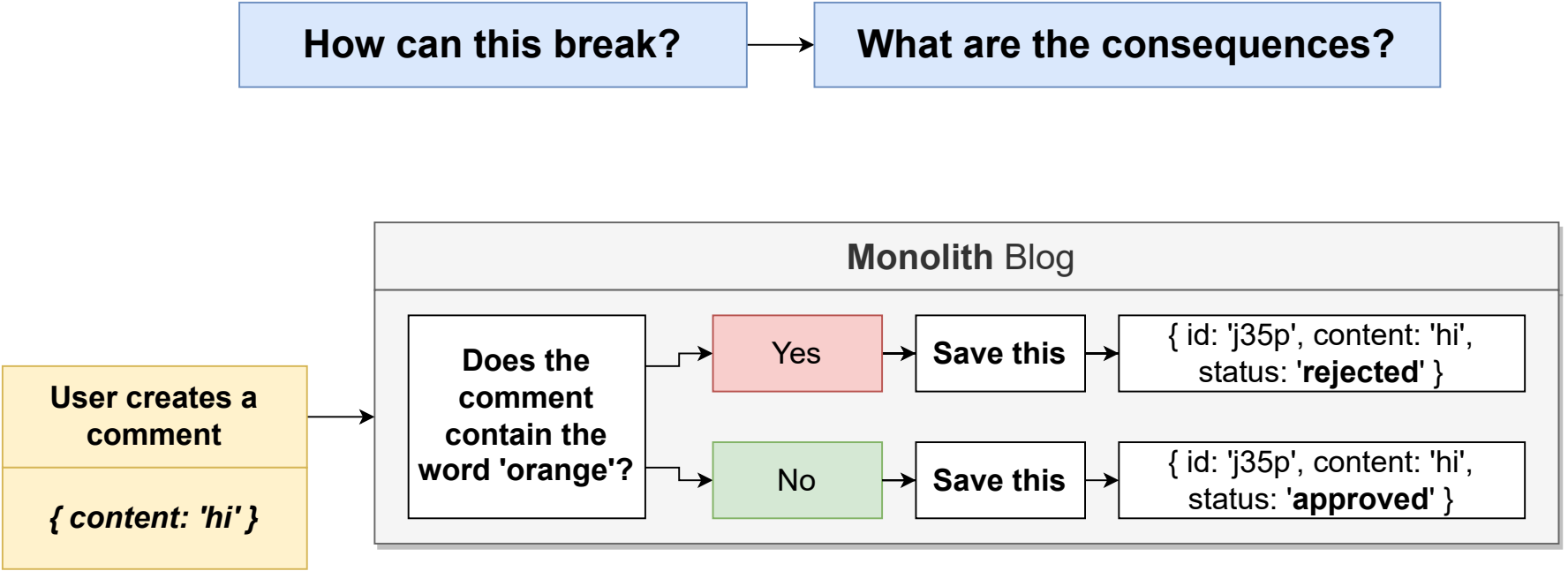
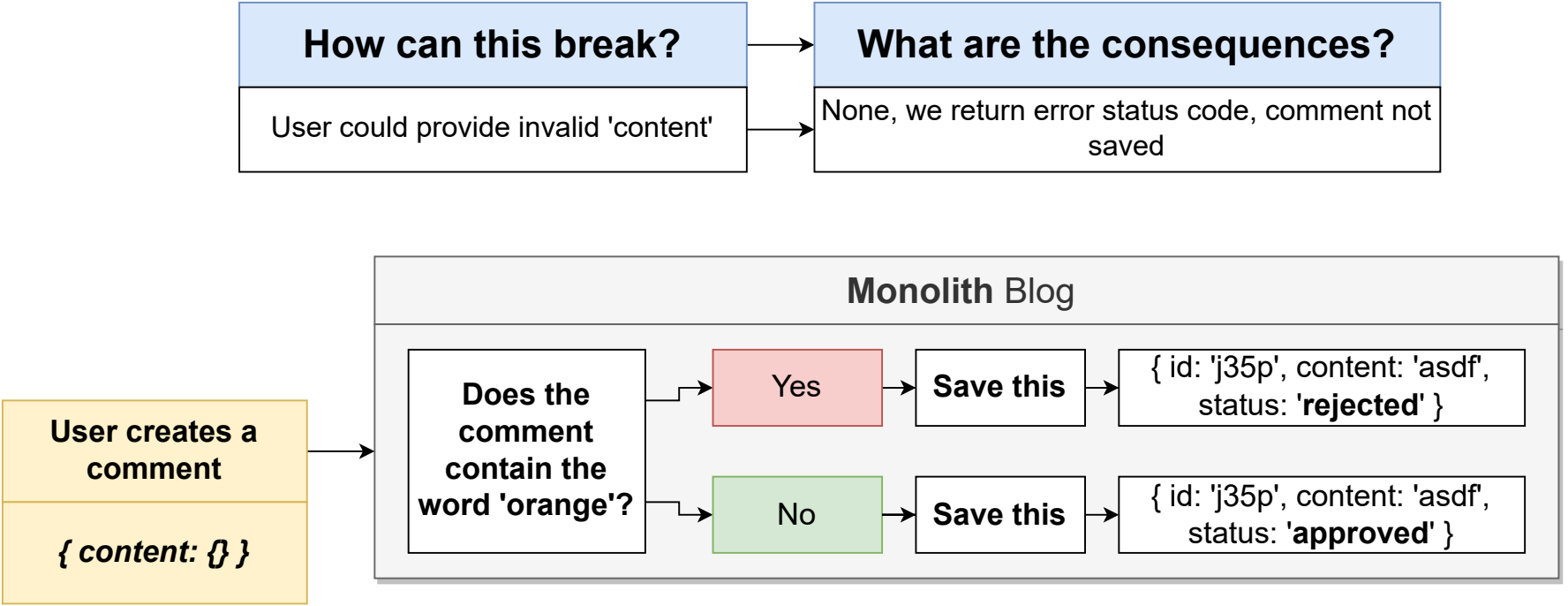
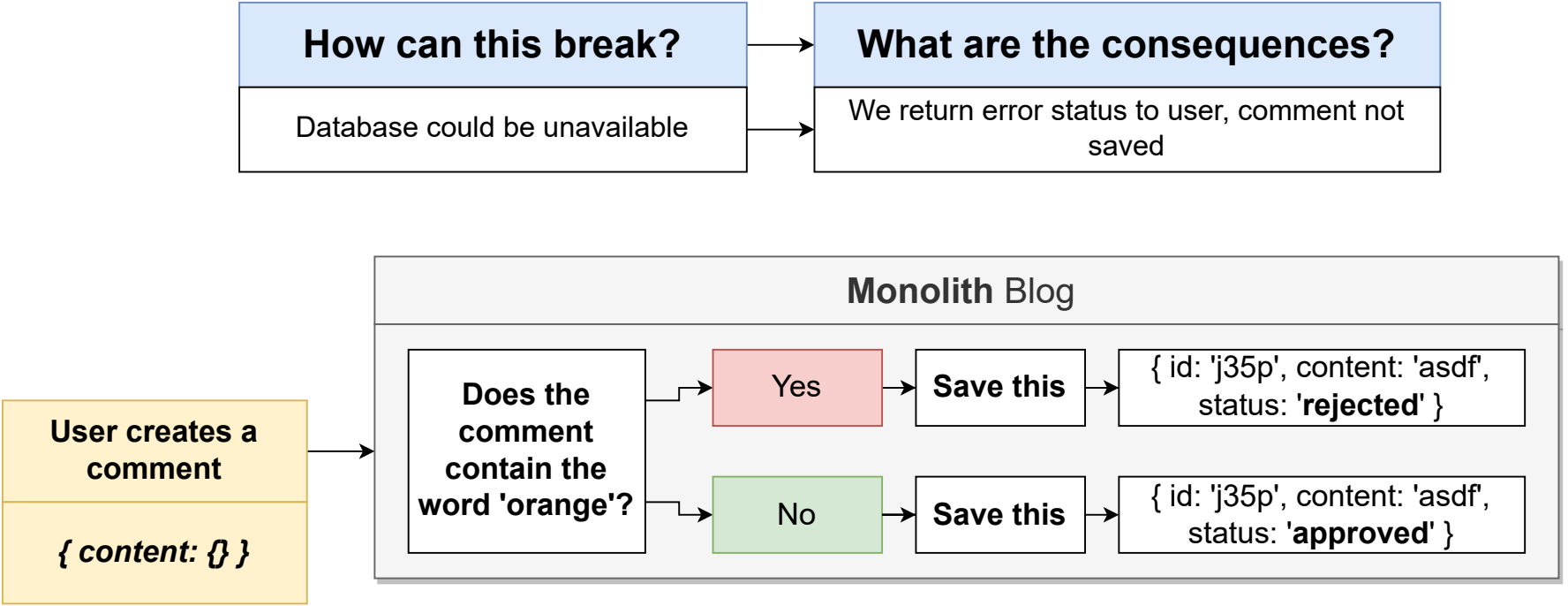
Finding Issues

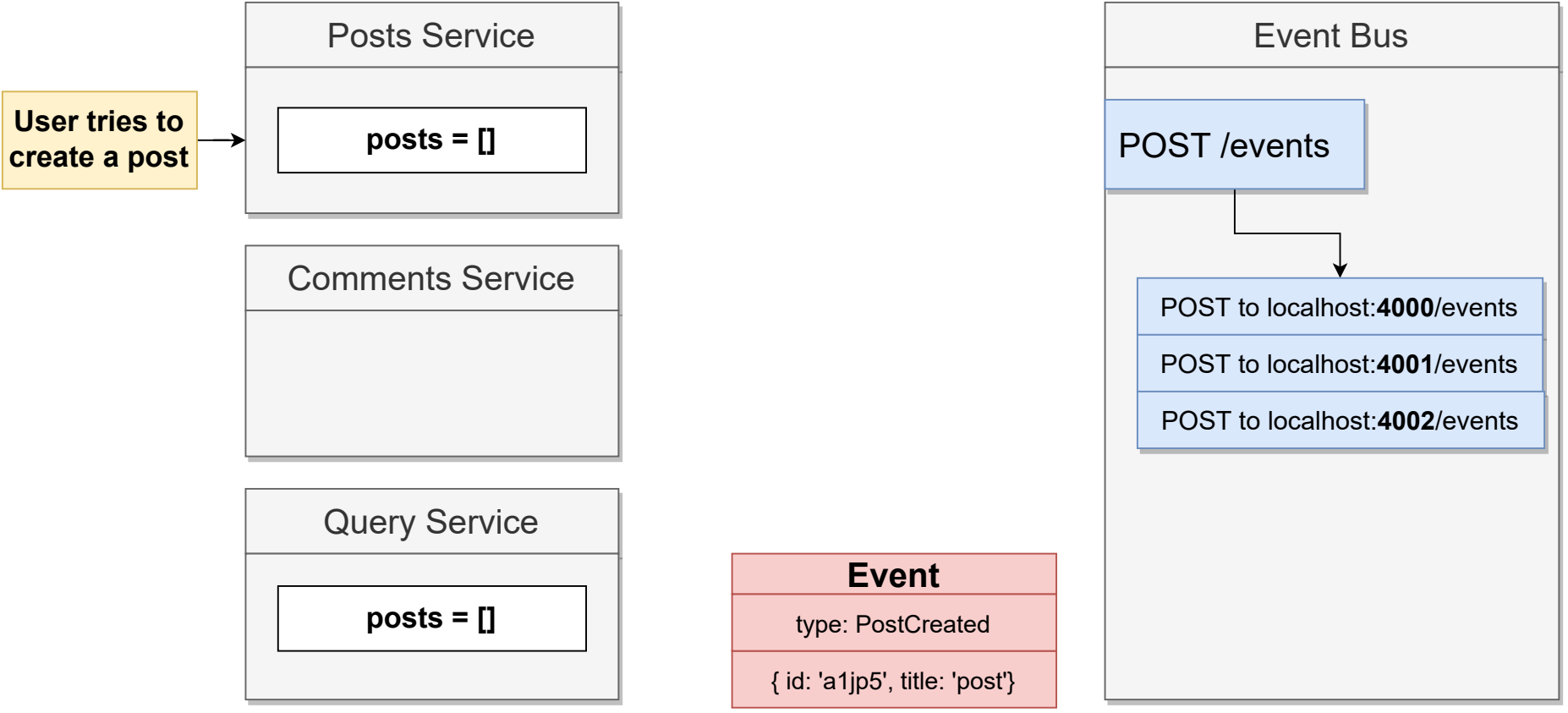


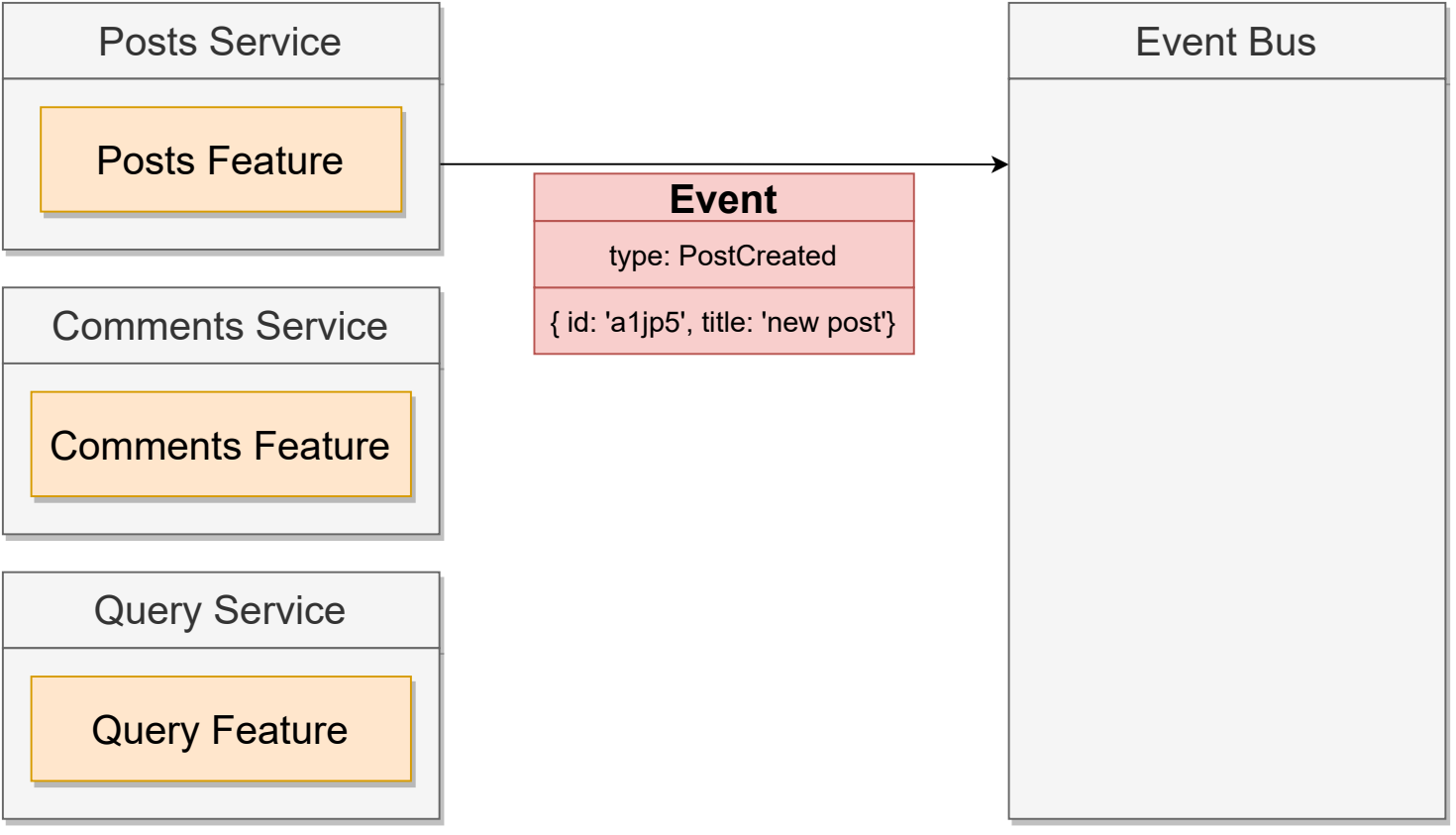

Monolith Blog
User creates a comment
Does the comment contain the word 'orange'?
Yes
No
Save this
Save this
{ id: 'j35p', content: 'asdf', status: 'rejected' }
{ id: 'j35p', content: 'asdf', status: 'approved' }


How can this break?
What are the consequences?
Monolith Blog
User creates a comment
{ content: 'hi' }
Does the comment contain the word 'orange'?
Yes
Save this
{ id: 'j35p', content: 'hi', status: 'rejected' }
No
Save this
{ id: 'j35p', content: 'hi', status: 'approved' }


How can this break?
What are the consequences?
User could provide invalid 'content'
None, we return error status code, comment not saved
Monolith Blog
User creates a comment
{ content: {} }
Does the comment contain the word 'orange'?
Yes
No
Save this
Save this
{ id: 'j35p', content: 'asdf', status: 'rejected' }
{ id: 'j35p', content: 'asdf', status: 'approved' }


How can this break?
What are the consequences?
Database could be unavailable
We return error status to user, comment not saved
Monolith Blog
User creates a comment
{ content: {} }
Does the comment contain the word 'orange'?
Yes
No
Save this
Save this
{ id: 'j35p', content: 'asdf', status: 'rejected' }
{ id: 'j35p', content: 'asdf', status: 'approved' }


Posts Service
User tries to create a post
posts = []
Comments Service
Query Service
posts = []
Event
type: PostCreated
{ id: 'a1jp5', title: 'post'}
Event Bus
POST /events
POST to localhost:4000/events
POST to localhost:4001/events
POST to localhost:4002/events


Posts Service
Posts Feature
Comments Service
Comments Feature
Query Service
Query Feature
Event
type: PostCreated
{ id: 'a1jp5', title: 'new post'}
Event Bus

Event Bus

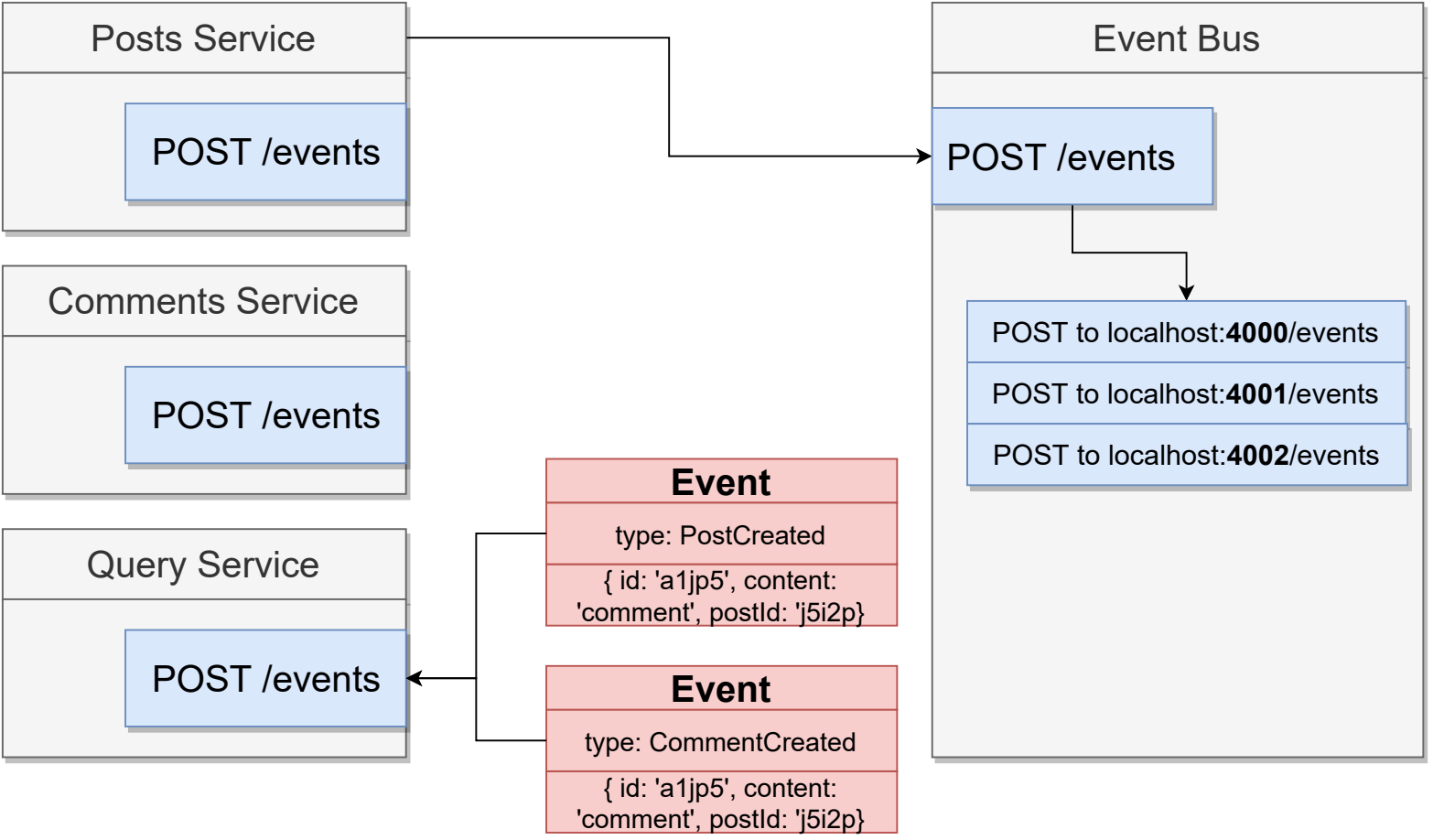
Many different implementations. RabbitMQ, Kafka, NATS...

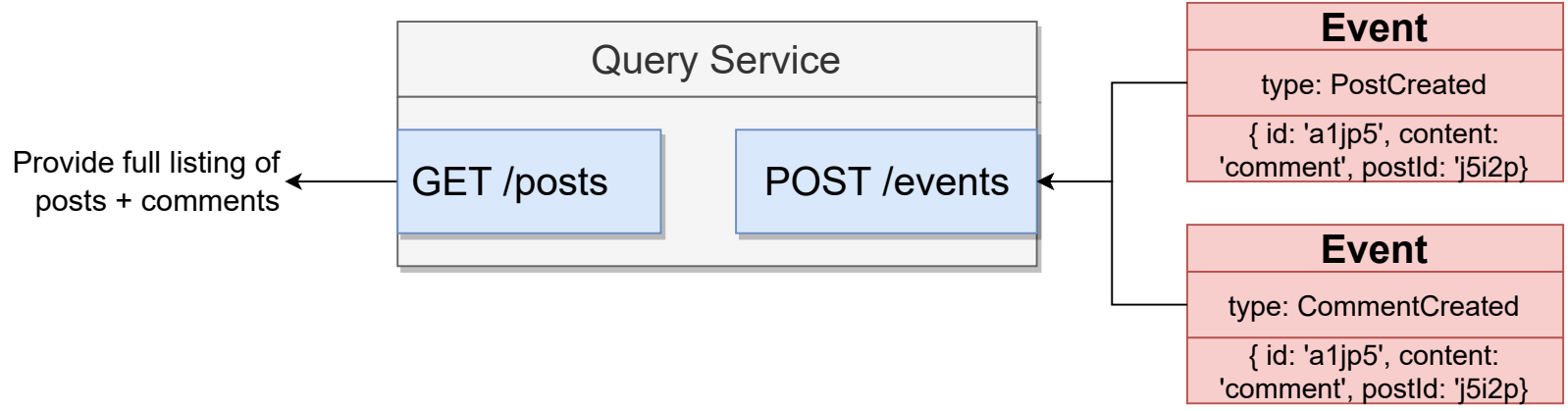
Receives events, publishes them to listeners

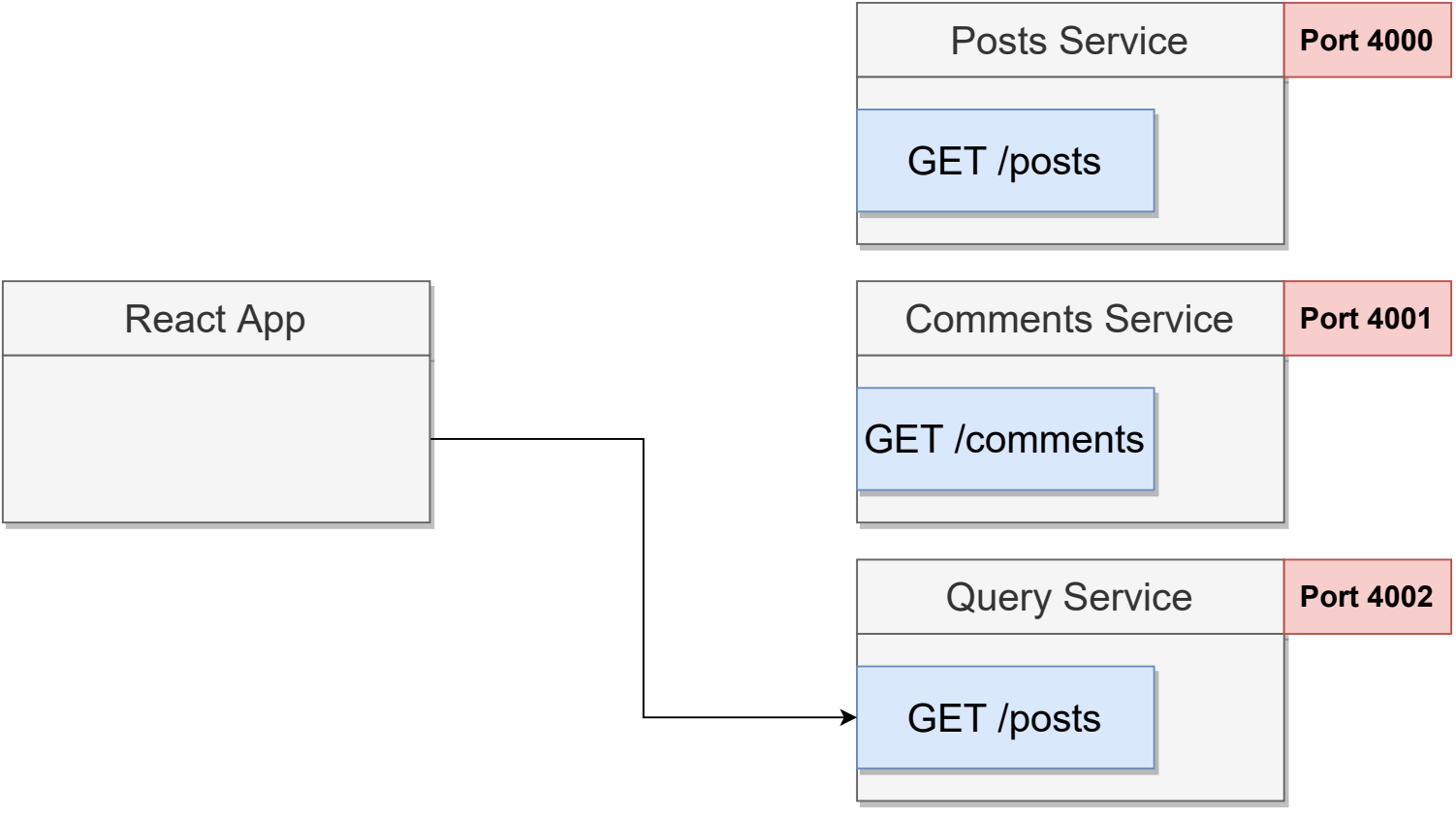
Many different subtle features that make async communication way easier or way harder

We are going to build our own event bus using Express. It will not implement the vast majority of features a normal bus has.

Yes, for our next app we will use a production grade, open source event bus


Posts Service
POST /events
Comments Service
POST /events
Query Service
POST /events
Event Bus
POST /events
POST to localhost:4000/events
POST to localhost:4001/events
POST to localhost:4002/events
Event
type: PostCreated
{ id: 'a1jp5', content: 'comment', postId: 'j5i2p}
Event
type: CommentCreated
{ id: 'a1jp5', content: 'comment', postId: 'j5i2p}


Query Service
GET /posts
POST /events
Provide full listing of posts + comments
Event
type: PostCreated
{ id: 'a1jp5', content: 'comment', postId: 'j5i2p}
Event
type: CommentCreated
{ id: 'a1jp5', content: 'comment', postId: 'j5i2p}


Posts Service
Port 4000
GET /posts
React App
Comments Service
Port 4001
GET /comments
Query Service
Port 4002
GET /posts

Add in comment moderation. Flag comments that contain the word 'orange'.

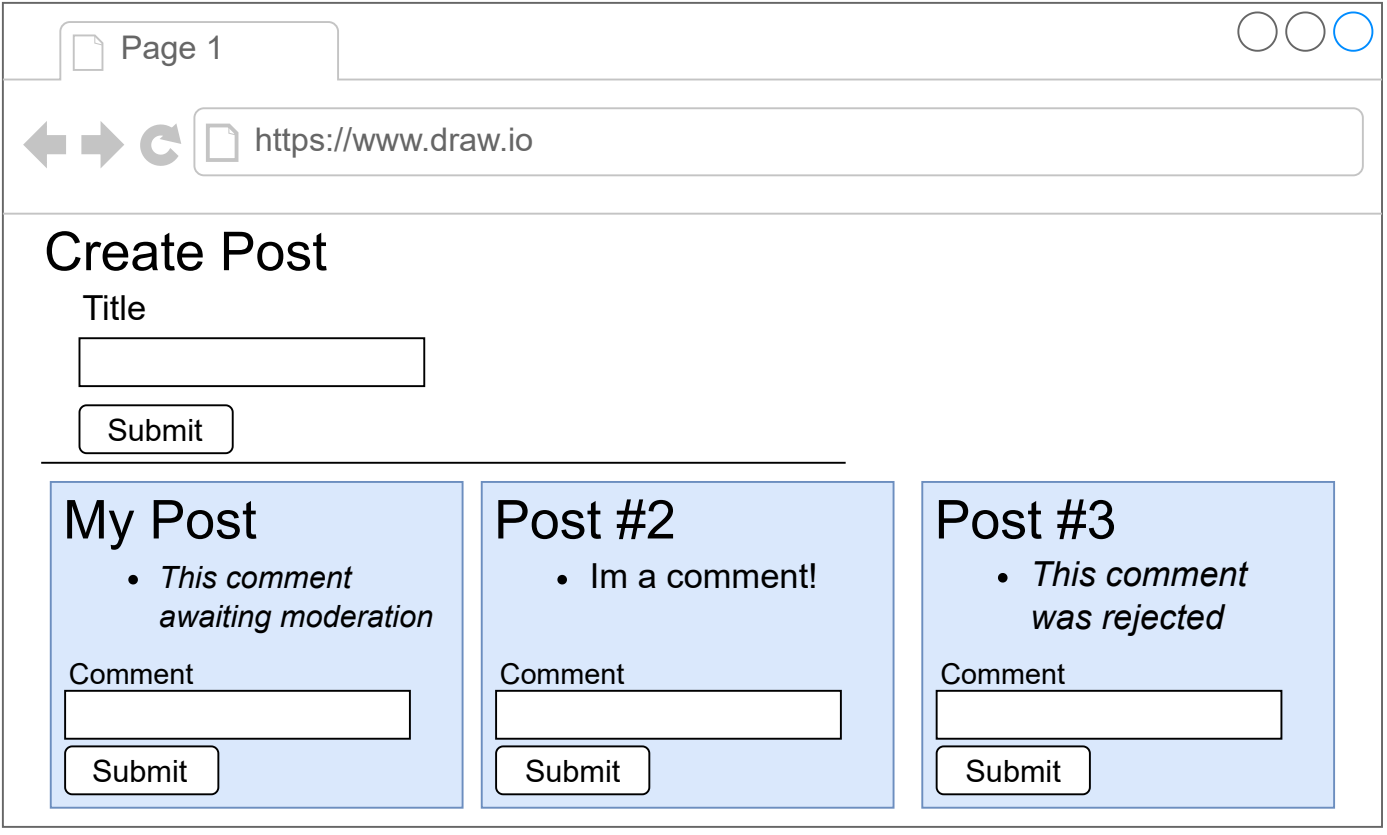
Feature Request

Feature Clarifications

Super easy to implement in the React app, but not if the filter list changes frequently

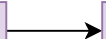
Super easy to implement in the existing comments service, but let's assume we want to add a new service

It might take a long time for the new service to moderate a comment.

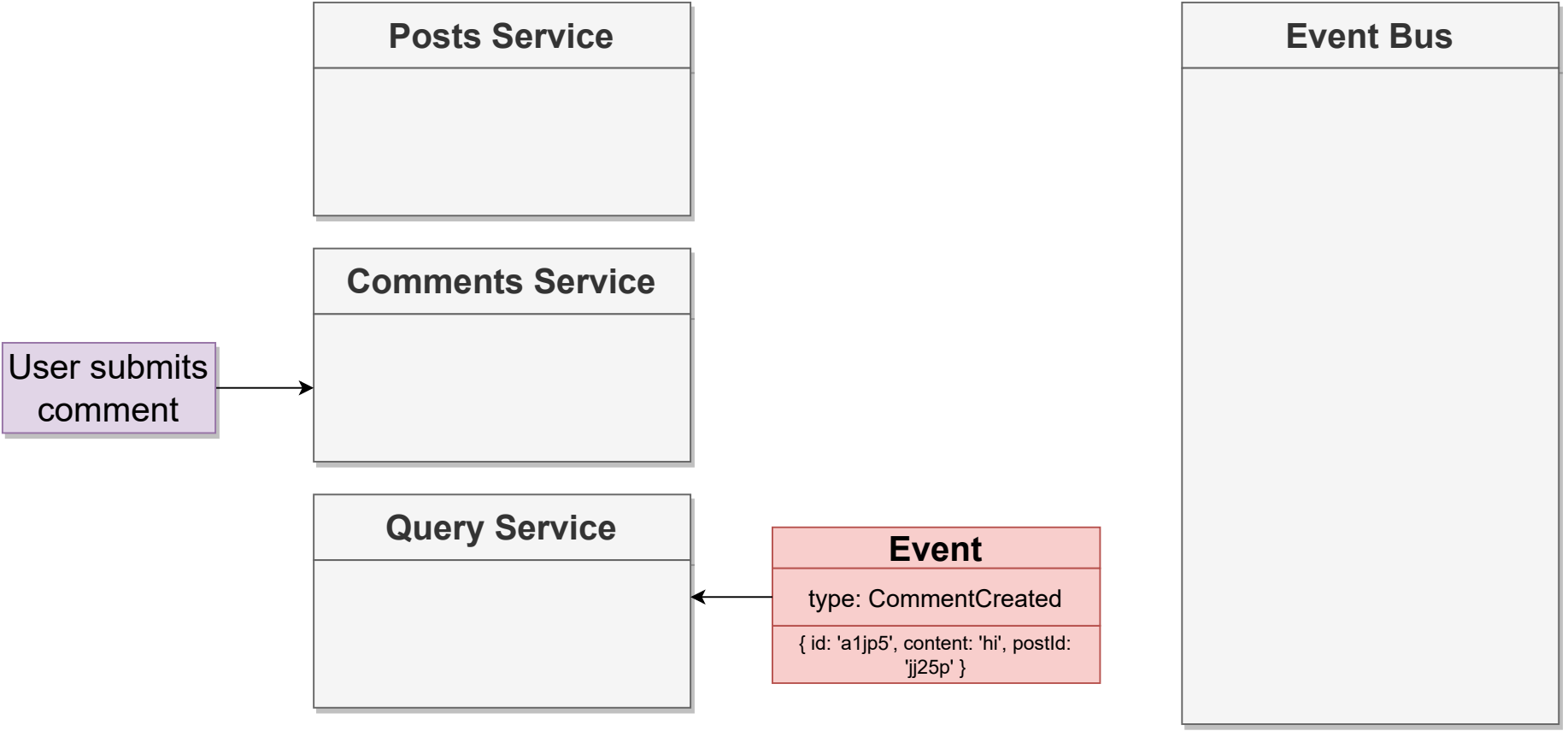

Page 1
https://www.draw.io
Create Post
Title
Submit
My Post
This comment awaiting moderation
Comment
Submit
Post #2
Im a comment!
Comment
Submit
Post #3
This comment was rejected
Comment
Submit

The React app needs to tell the difference between a comment that is pending moderation, rejected, or approved

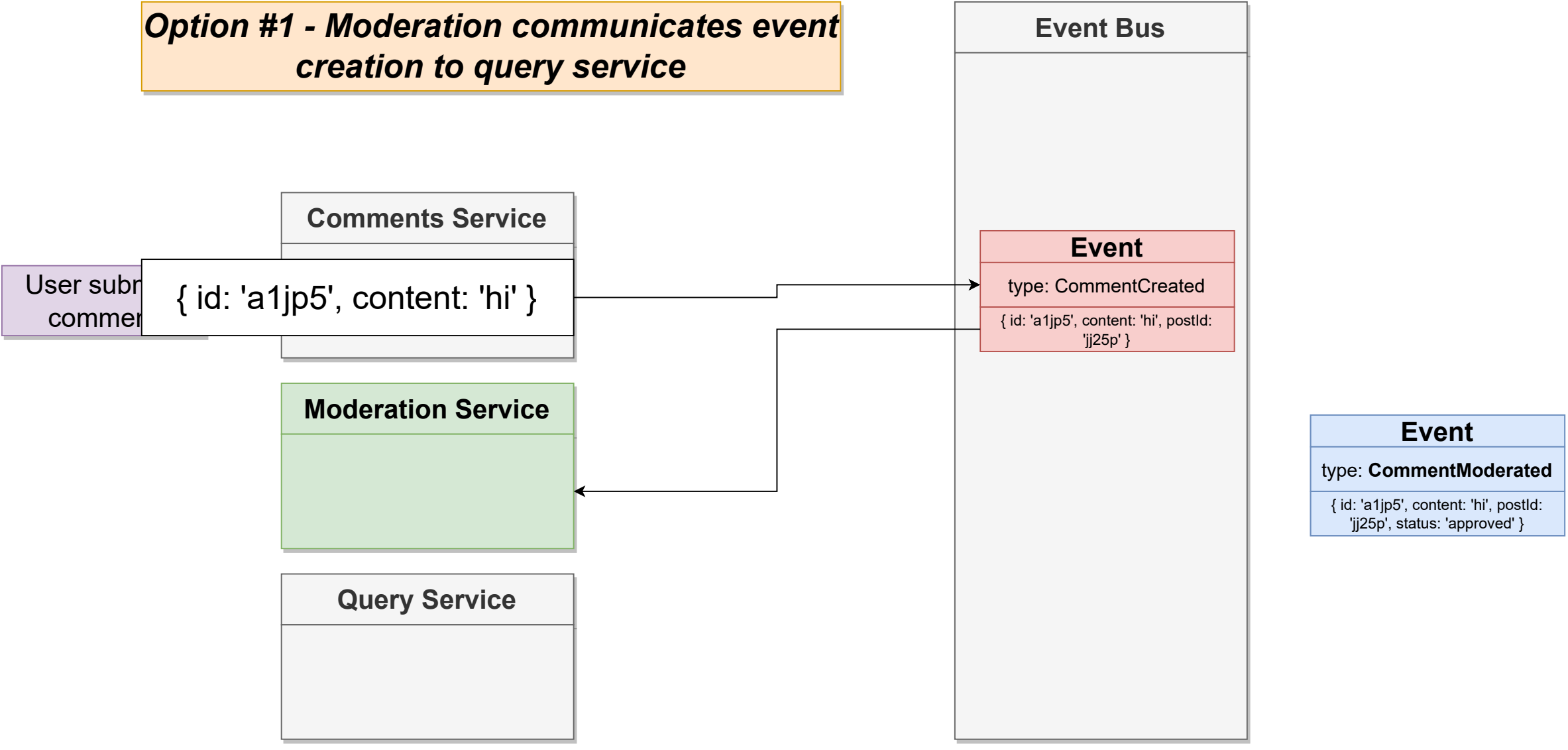
Comment	
name	type
id	string
content	string

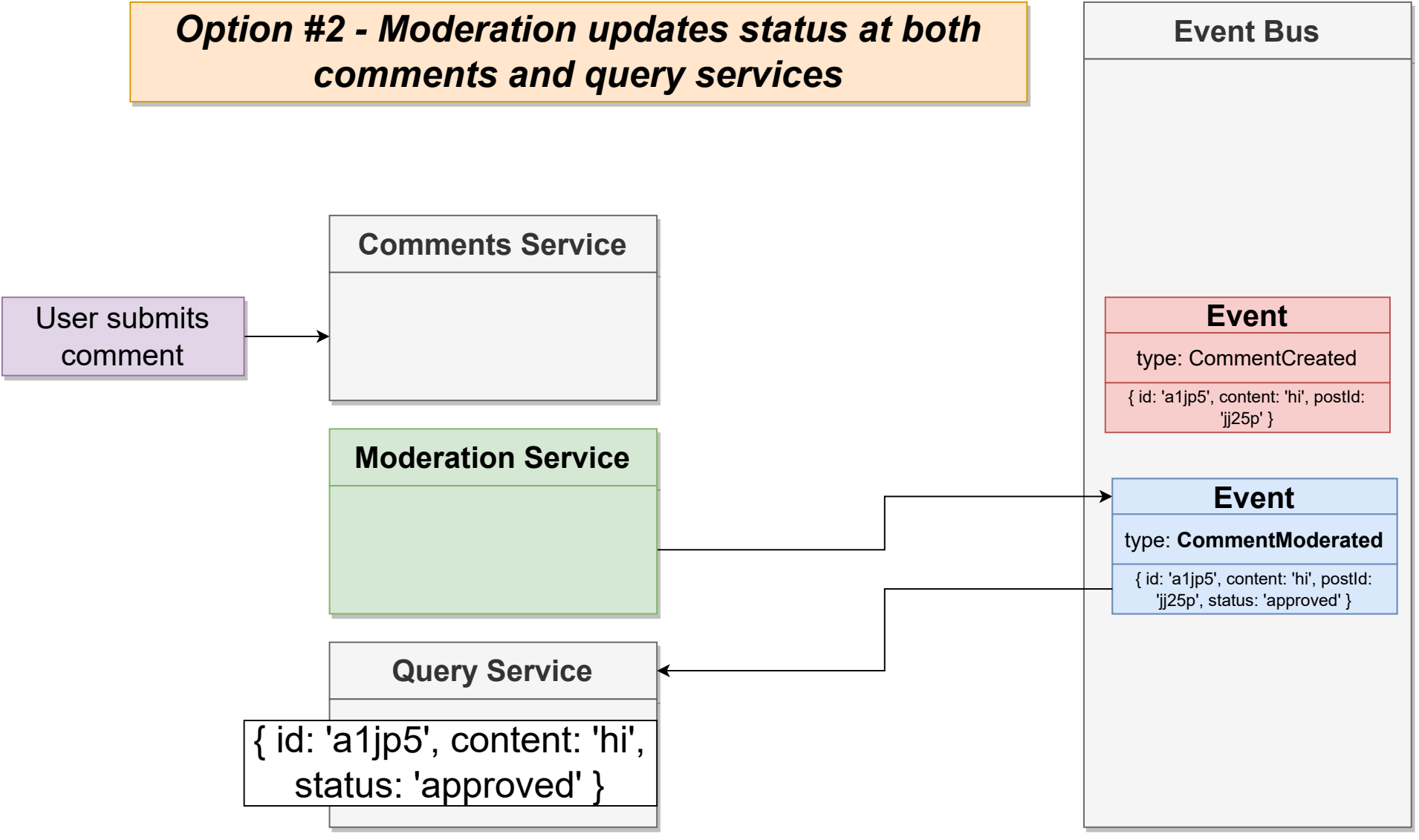


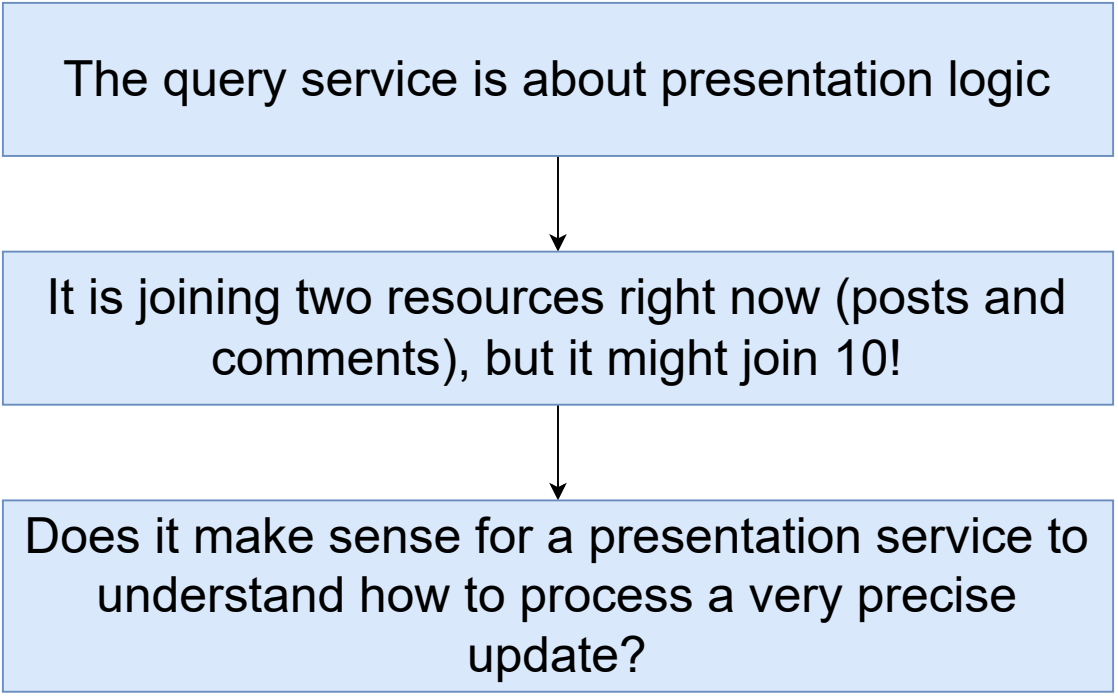
Comment	
name	type
id	string
content	string
status	'approved' | 'rejected' | 'pending

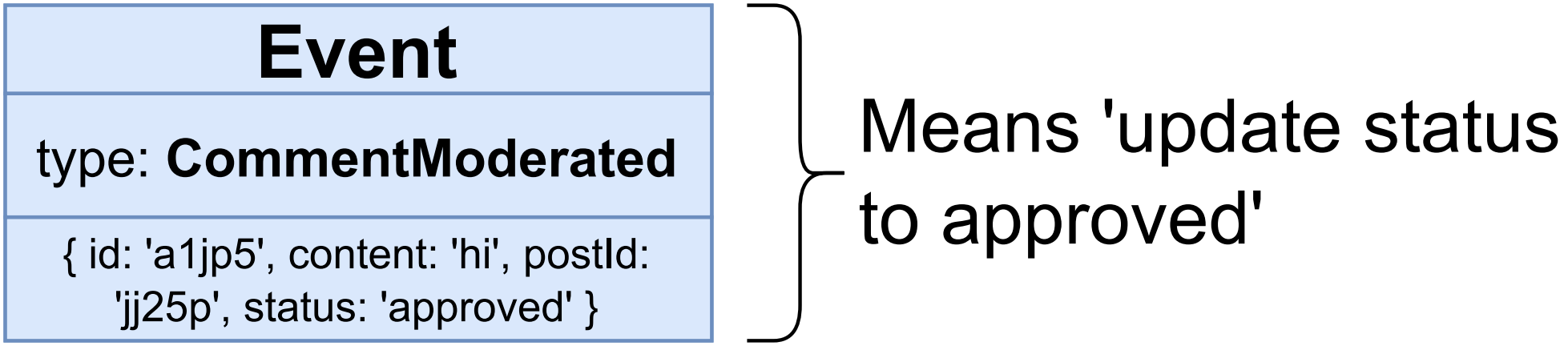

Posts Service
Comments Service
User submits comment
Query Service
Event
type: CommentCreated
{ id: 'a1jp5', content: 'hi', postId: 'jj25p' }
Event Bus

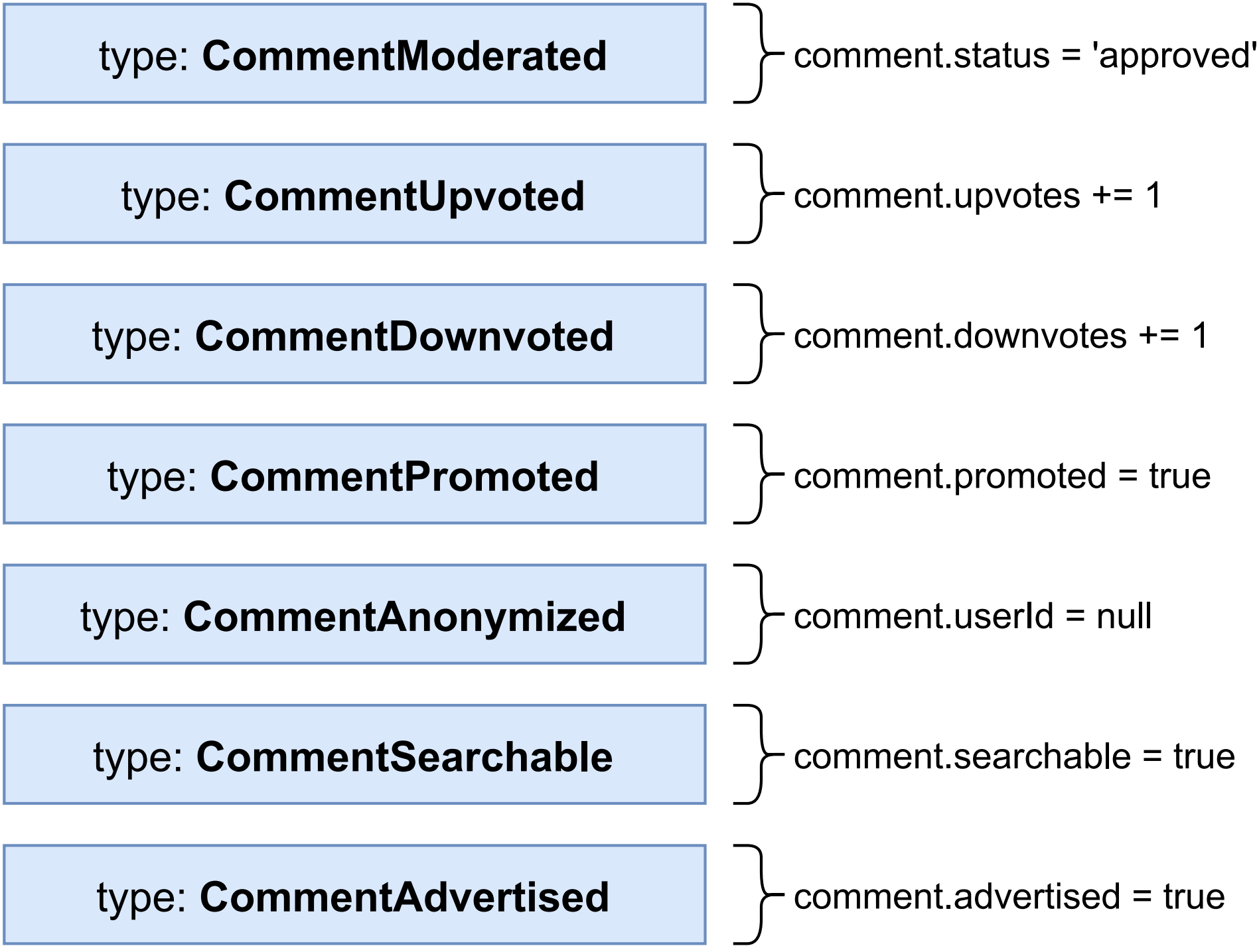
Option #1 - Moderation communicates event creation to query service

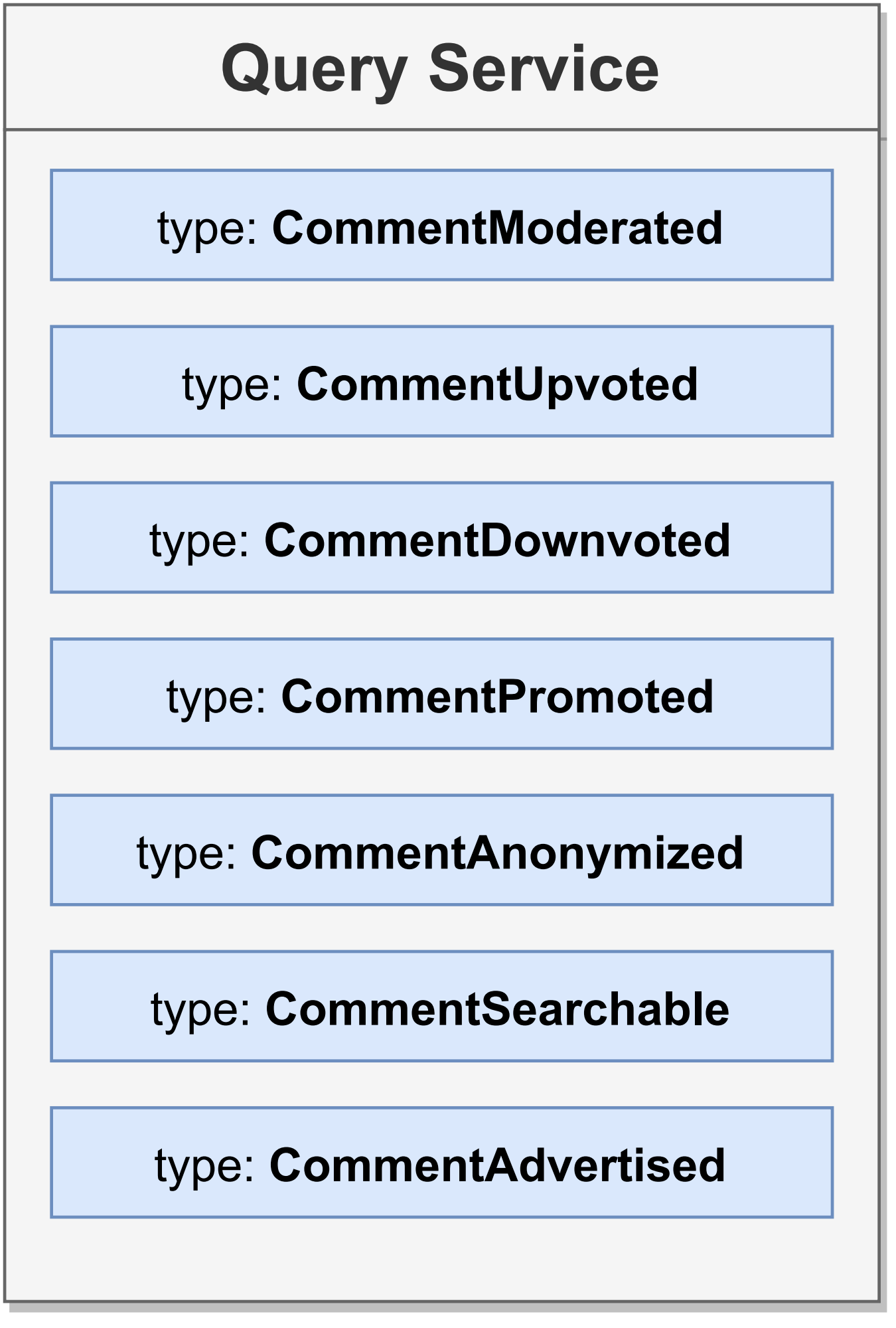


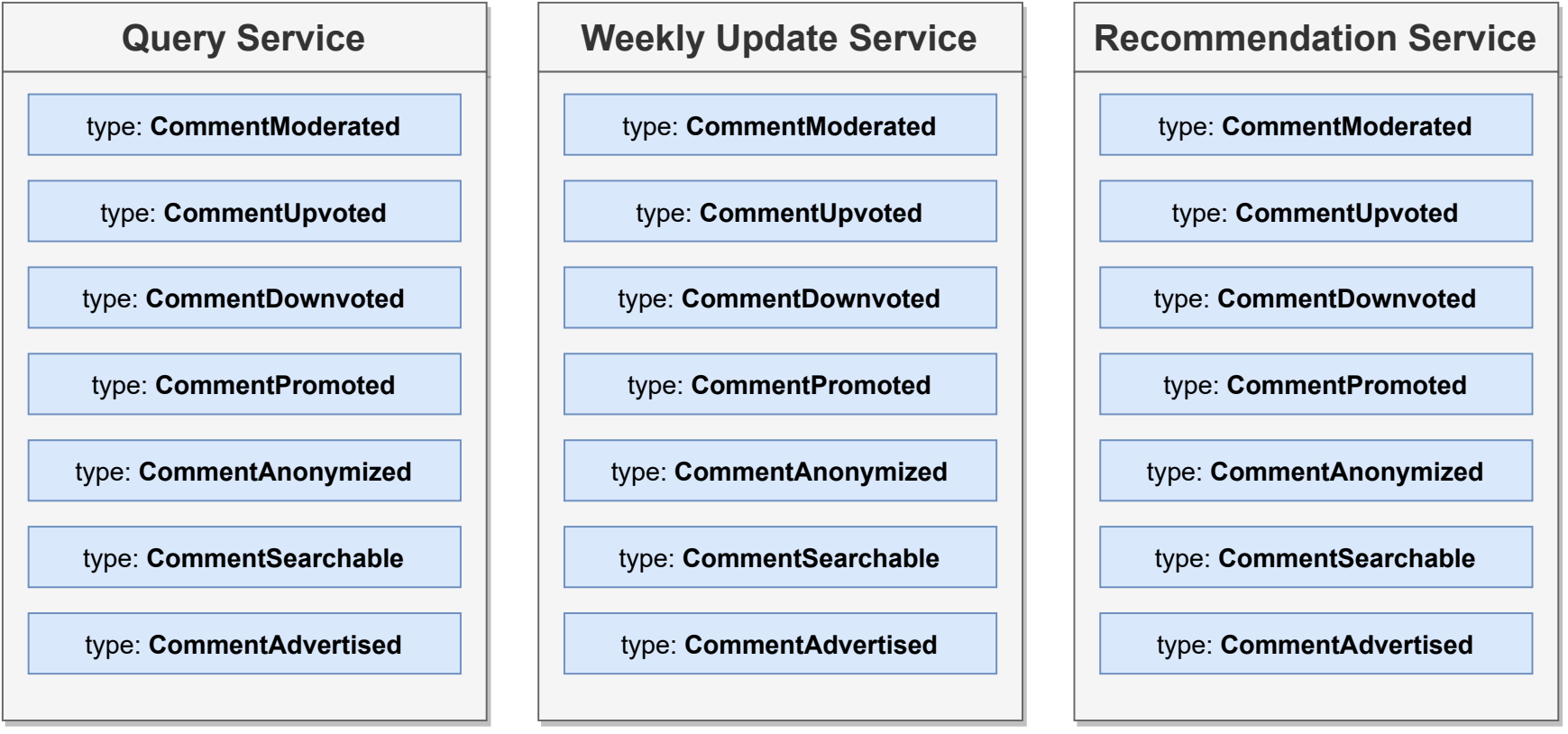

Option #2 - Moderation updates status at both comments and query services
Event Bus
Comments Service
User submits comment
Event
type: CommentCreated
{ id: 'a1jp5', content: 'hi', postId: 'jj25p' }
Moderation Service
Event
type: CommentModerated
{ id: 'a1jp5', content: 'hi', postId: 'jj25p', status: 'approved' }
Query Service
{ id: 'a1jp5', content: 'hi', status: 'approved' }

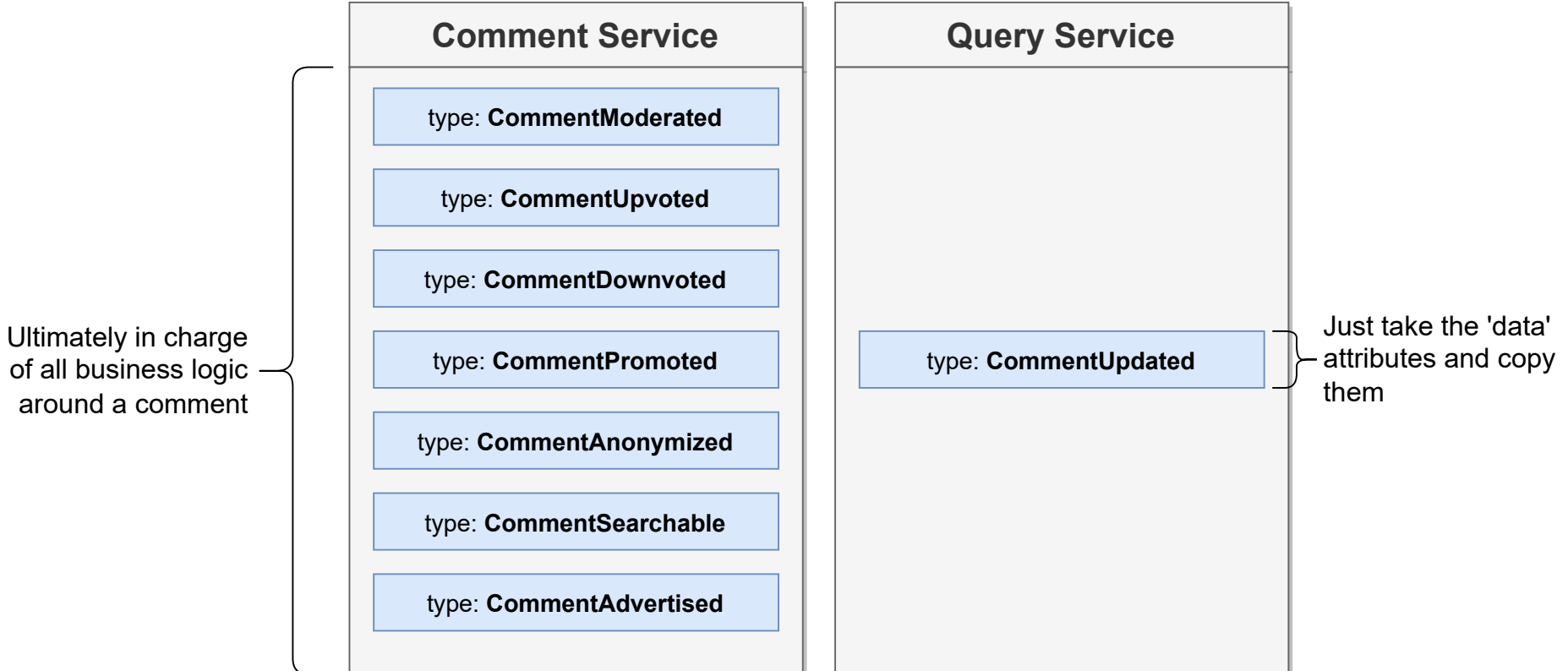

The query service is about presentation logic
It is joining two resources right now (posts and comments), but it might join 10!
Does it make sense for a presentation service to understand how to process a very precise update?

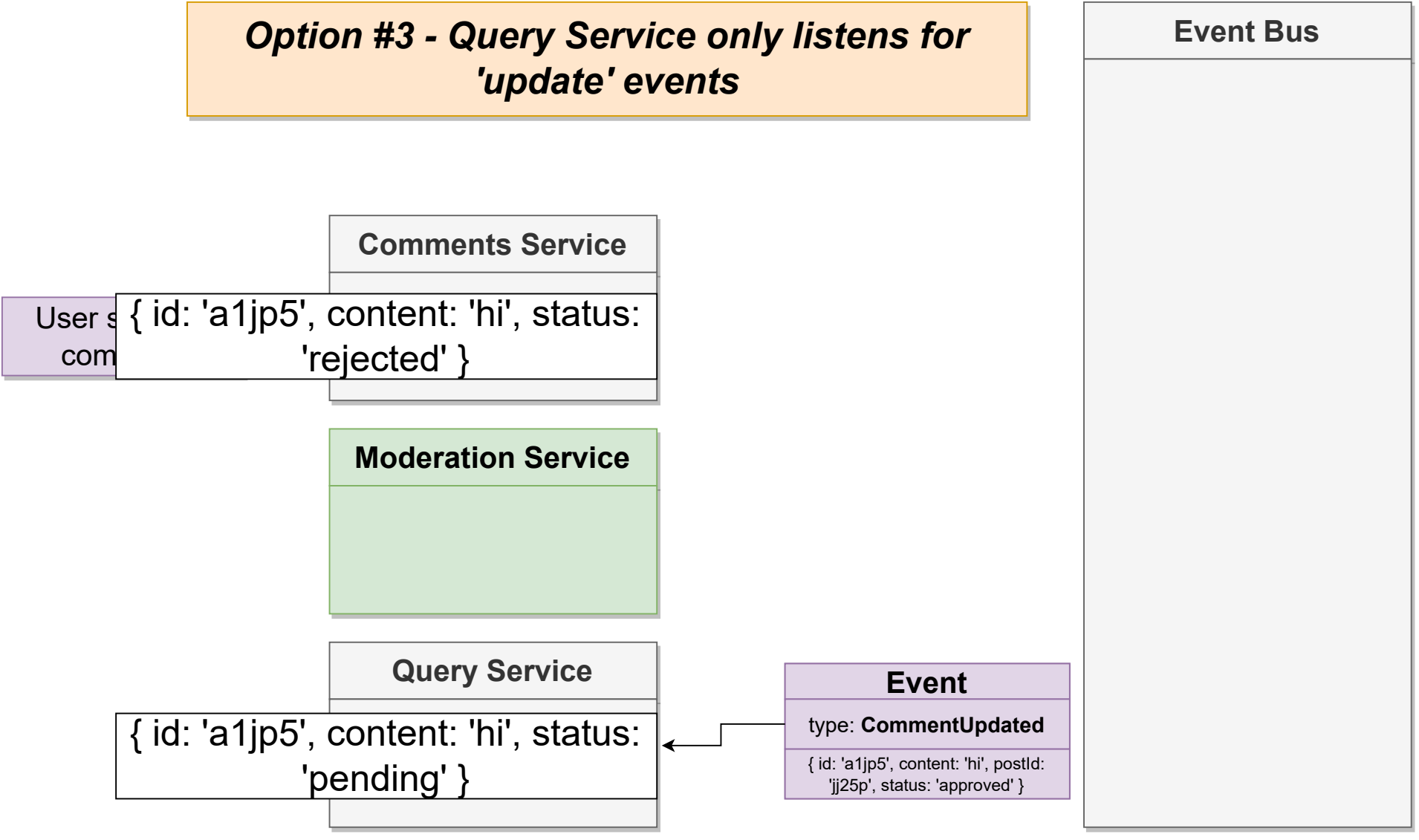

Event
type: CommentModerated
{ id: 'a1jp5', content: 'hi', postId: 'jj25p', status: 'approved' }
Means 'update status to approved'


type: **CommentModerated**
comment.status = 'approved'
type: **CommentUpvoted**
comment.upvotes += 1
type: **CommentDownvoted**
comment.downvotes += 1
type: **CommentPromoted**
comment.promoted = true
type: **CommentAnonymized**
comment.userId = null
type: **CommentSearchable**
comment.searchable = true
type: **CommentAdvertised**
comment.advertised = true


Query Service
type: CommentModerated
type: CommentUpvoted
type: CommentDownvoted
type: CommentPromoted
type: CommentAnonymized
type: CommentSearchable
type: CommentAdvertised

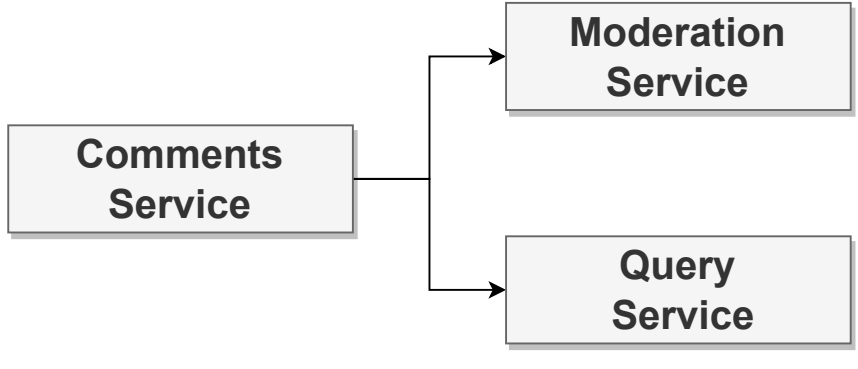

Query Service
type: CommentModerated
type: CommentUpvoted
type: CommentDownvoted
type: CommentPromoted
type: CommentAnonymized
type: CommentSearchable
type: CommentAdvertised
Weekly Update Service
type: CommentModerated
type: CommentUpvoted
type: CommentDownvoted
type: CommentPromoted
type: CommentAnonymized
type: CommentSearchable
type: CommentAdvertised
Recommendation Service
type: CommentModerated
type: CommentUpvoted
type: CommentDownvoted
type: CommentPromoted
type: CommentAnonymized
type: CommentSearchable
type: CommentAdvertised


Comment Service
type: CommentModerated
type: CommentUpvoted
type: CommentDownvoted
type: CommentPromoted
type: CommentAnonymized
type: CommentSearchable
type: CommentAdvertised
Query Service
type: CommentUpdated
Ultimately in charge of all business logic around a comment
Just take the 'data' attributes and copy them


Option #3 - Query Service only listens for 'update' events
Event Bus
Comments Service
User s
com
{ id: 'a1jp5', content: 'hi', status: 'rejected' }
Moderation Service
Query Service
{ id: 'a1jp5', content: 'hi', status: 'pending' }
Event
type: CommentUpdated
{ id: 'a1jp5', content: 'hi', postId: 'jj25p', status: 'approved' }

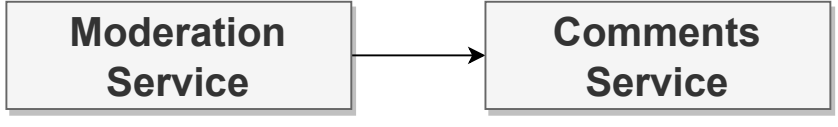
CommentCreated	
id	string
content	string
postId	string
status	'pending'

Emitted when a comment is created by the comments service



CommentModerated	
id	string
content	string
postId	string
status	'approved' | 'rejected'

Emitted when the moderation service moderates a comment



CommentUpdated	
id	string
content	string
postId	string
status	'approved' | 'rejected'

Emitted when the comment service updates attributes on a comment