

ToDo Case Study

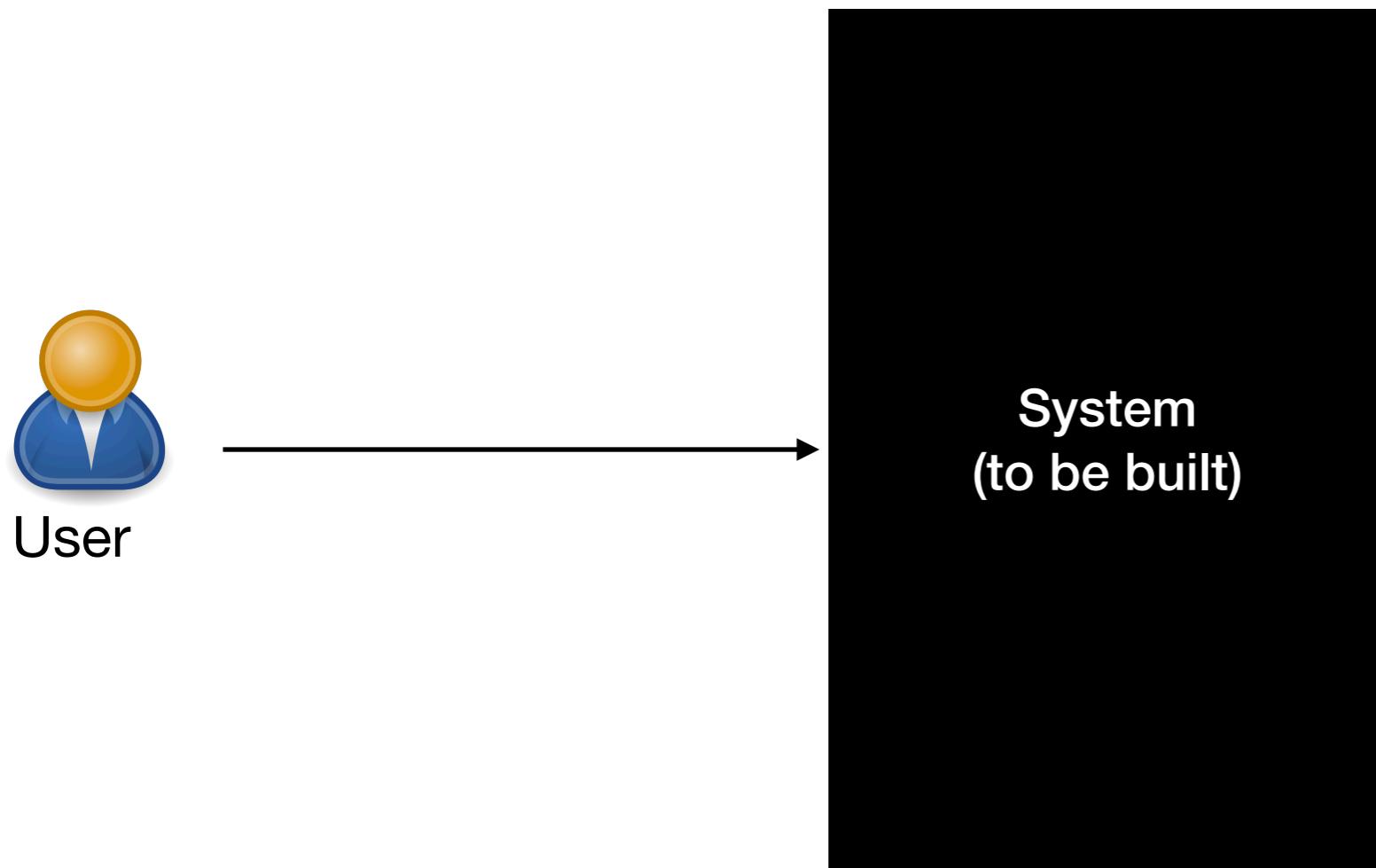
- Requirements
- Definition
- Eval

Architectural Requirements

- # Context View (understand the actors of the system)**
- # Functional View (understand major functionality)**
- # Constraints (any rules to be followed ?)**
- # Quality View (which quality should the system support ?)**

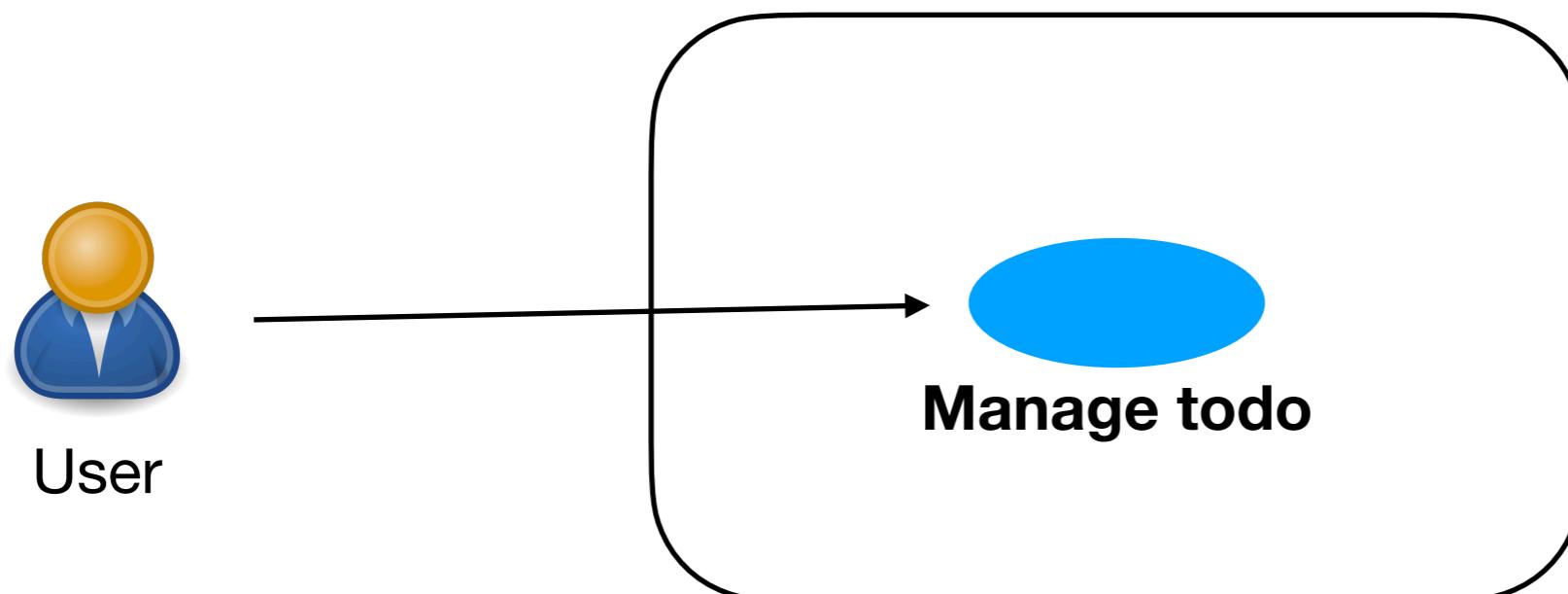
Context View

“Identify all actors”



Functional View

“Identify key functionality”



Constraints

“Identify rules imposed by stake holders”

- Should work from IE 11 browser

Quality Requirements

Source (who)	Stimulus (action)	Artifact (which)	Environment (context)	Response (output)	Measure (scale)
Someone of unknown identity who is external and not authorized with access to limited resources	requests to add a new Todo	"In the TodoWebApp"	"while it is online and running normally"	"The response is to block access to the data and record the access attempts"	"with 100% probability of detecting the attack, 100% probability of denying access, and 50% probability of identifying the location of the individual."

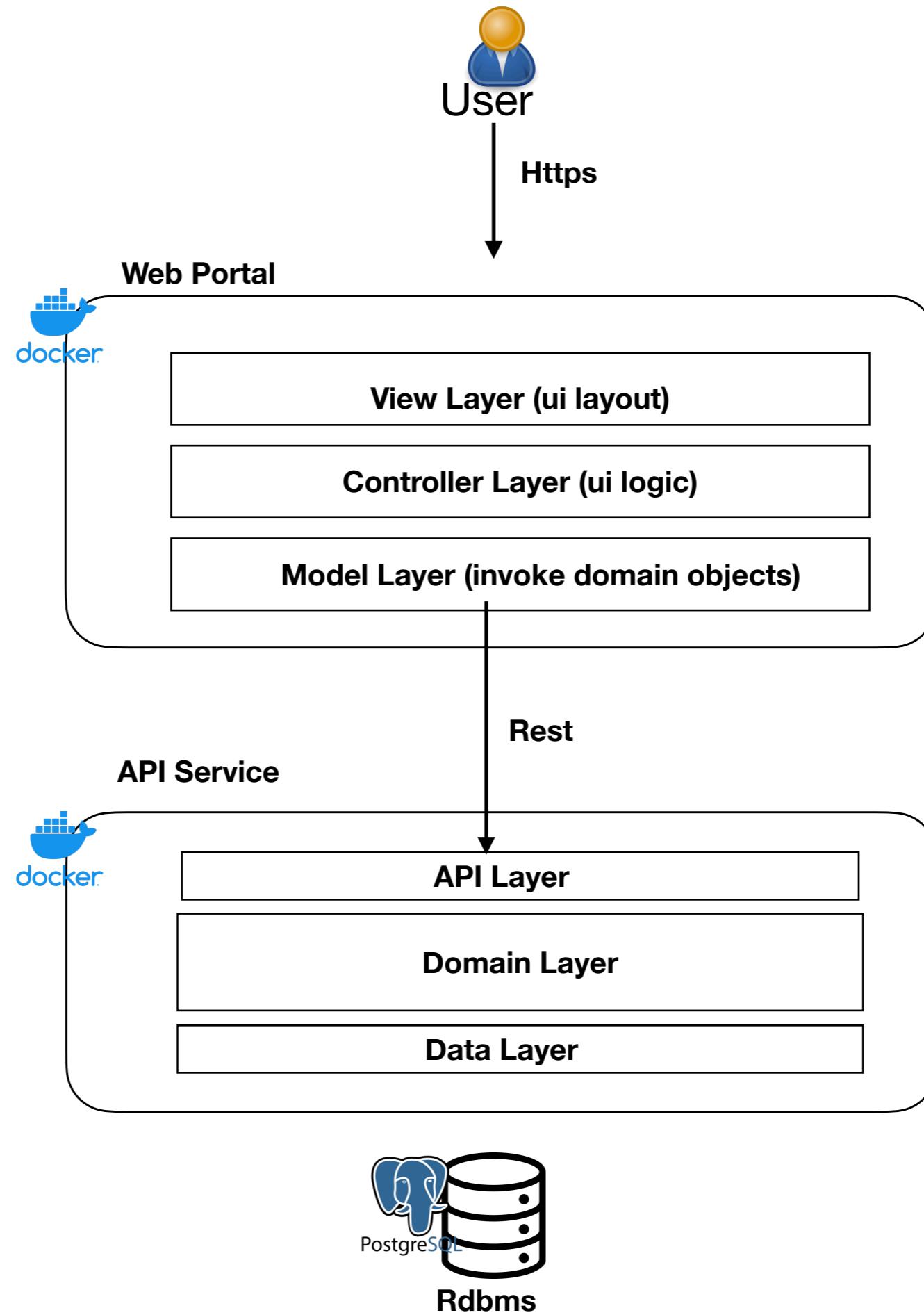
Architectural Design

- # Logical View (decomposition decisions)
- # Deployment View (deployment decisions)
- # Security View (Security decisions)
- # ...

Logical View

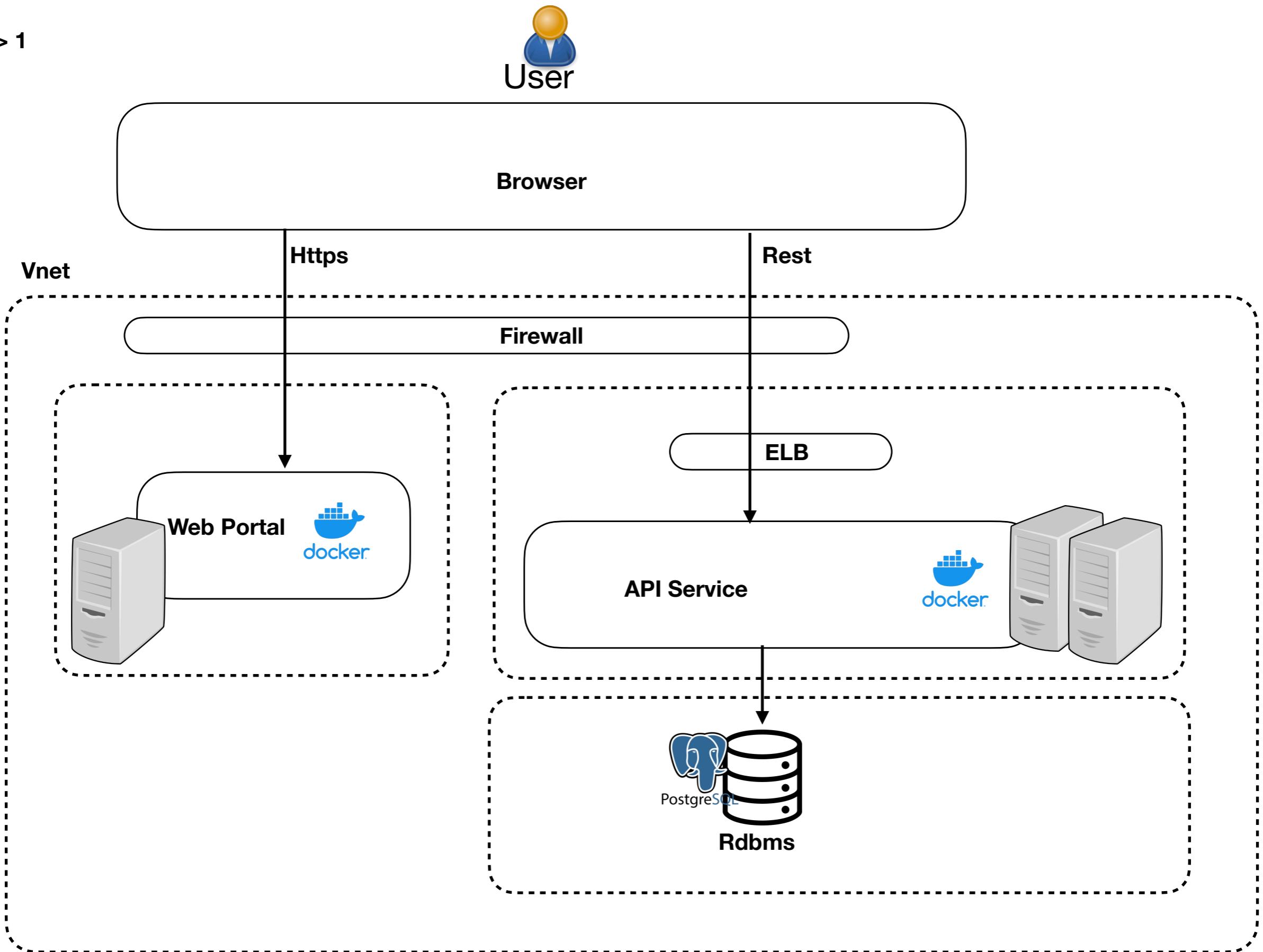
- # System Decomposition**
- # Persistence approach**
- # Compute approach**
- # Communication approach**
- # cross cutting approach**

4 -> 1



Deployment View

4 -> 1



Security View

Authentication (who are you)

centralize(oauth2) - Auth0

Authorization (who can do what)

centralize(claim) - Auth0

Audit (who did , what)

Audit log

Input validation

Asset Handling In rest (storage)

Asset Handling In Transit (wire)

Exception Handling (Code)

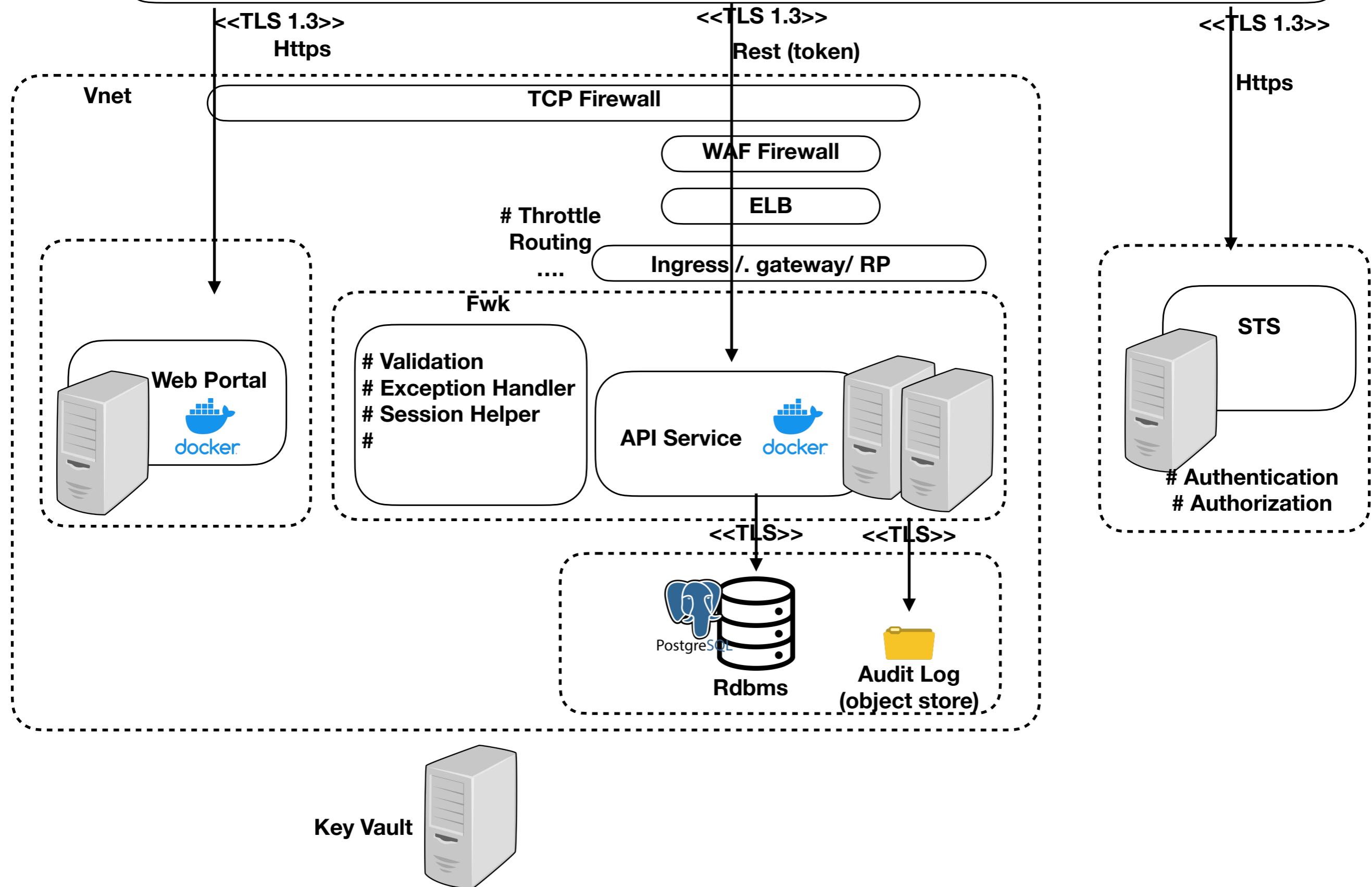
Session Handling (code)

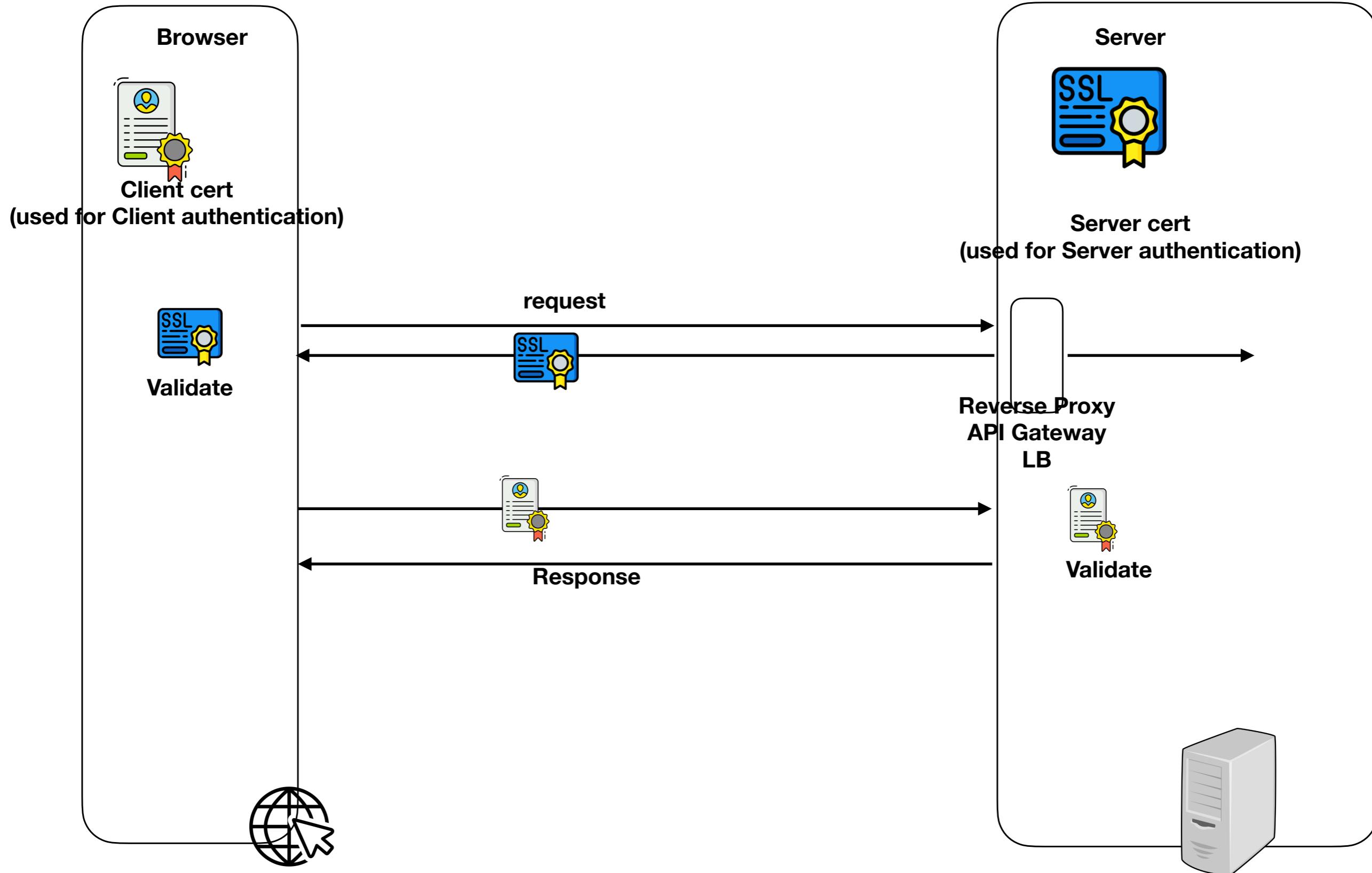
Key Management

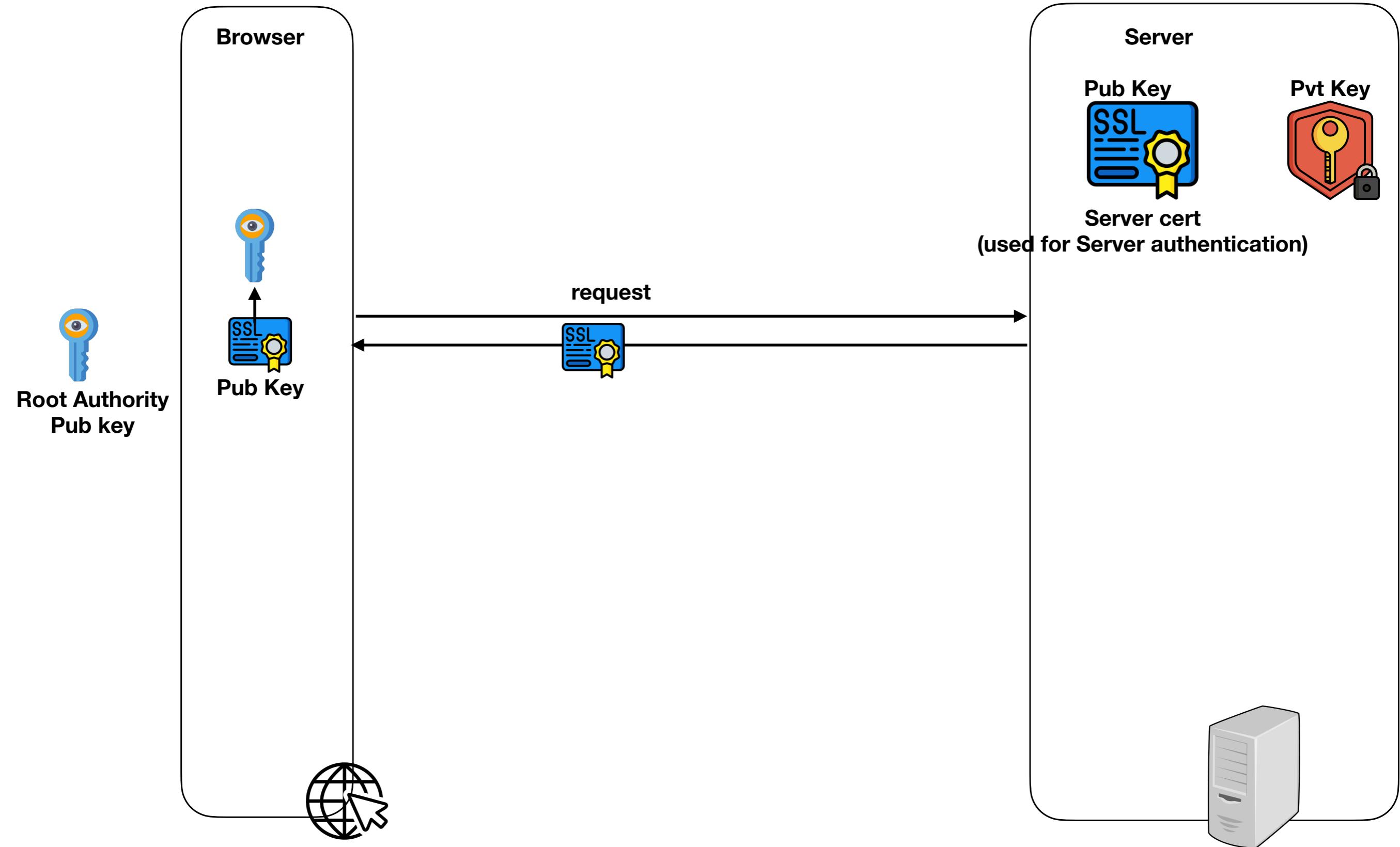
Throttling



Browser







Architectural Eval

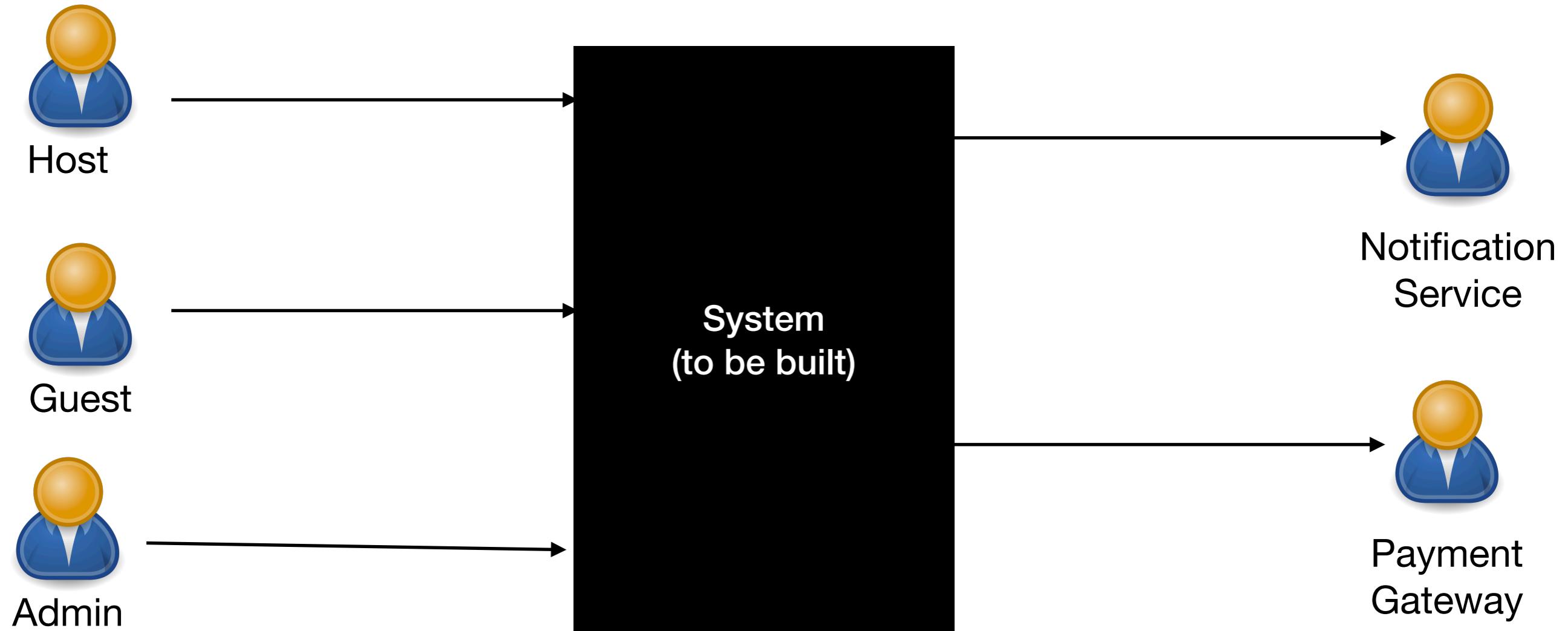
Airbnb Case Study

Architectural Requirements

- # Context View (understand the actors of the system)
- # Functional View (understand major functionality)
- # Constraints (any rules to be followed ?)
- # Quality View (which quality should the system support ?) ***
- # Assumptions (assumptions made by the architect)

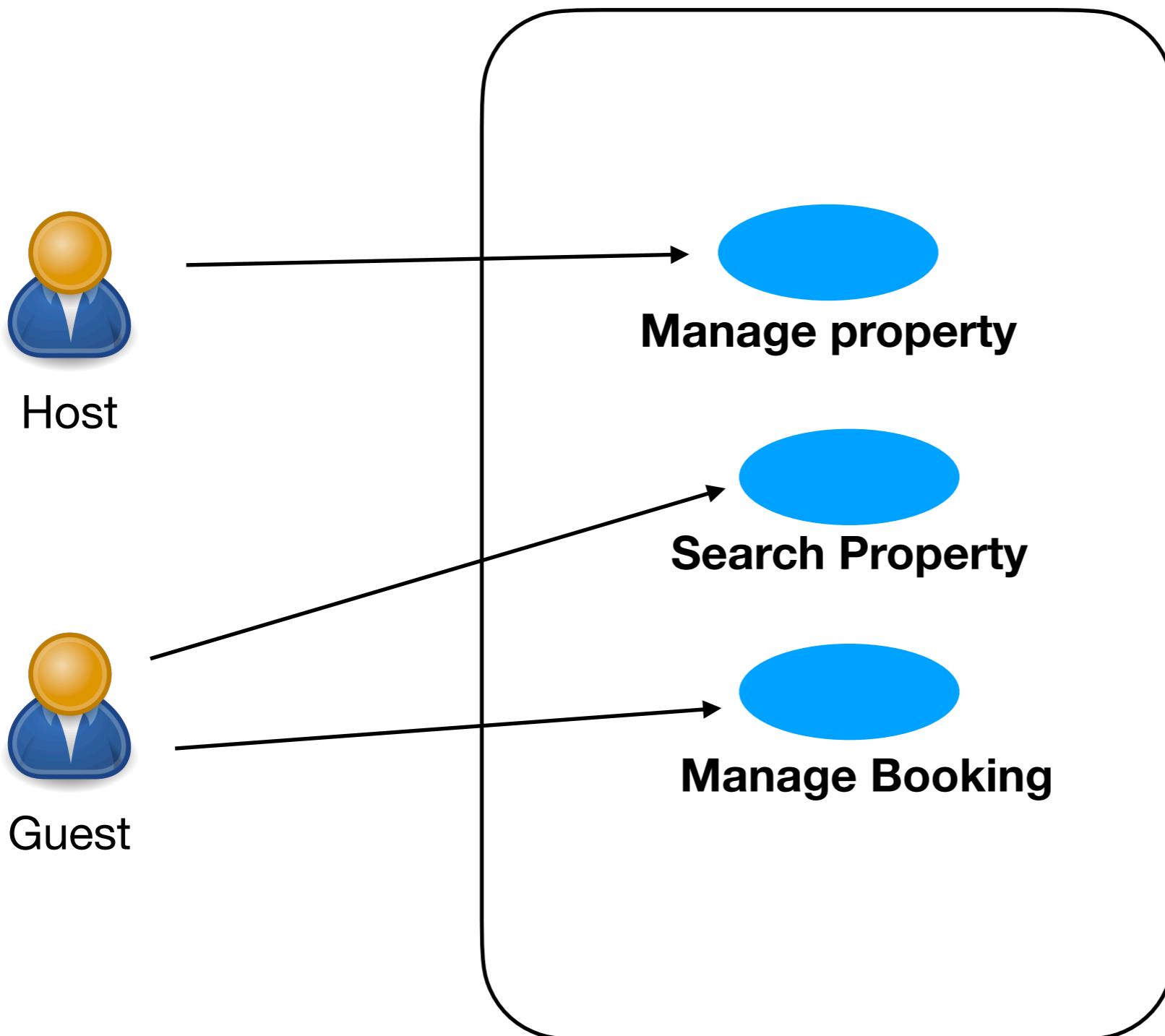
Context View

“Identify all actors”



Functional View

“Identify key functionality”



Constraints

“Identify rules imposed by stakeholders”

- Should not have affinity to a cloud vendor. Should be able to deploy it in Azure, Aws, GCP or on prem.
- Data should reside in the country where it data was created.
- Should support SCIM for Identity management.
-

Quality Requirements

Source (who)	Stimulus (action)	Artifact (which)	Environment (context)	Response (output)	Measure (scale)
A guest	Searching for a property	On the web portal	during peak hours	Should get property listing	In < 2 seconds.
A unknown identity	requests to add a property	In the Web Portal	during normal hours	The response is to block access to the data and record the access attempts	with 100% probability of detecting the attack, 100% probability of denying access, and 50% probability of identifying the location of the
A guest	Submits a booking	On the portal	Duplicate submit	The guest is not doubly charged	With a 100% probability of detecting the duplicate request
Developer	Want to add a new payment gateway	On the portal	During maintenance	The payment gateway is added	In < 2 man days
The Database	Failed	In the Data Centre	During Operational Hours	Secondary is made Primary	In < 2 minutes

Assumptions

“self made decisions”

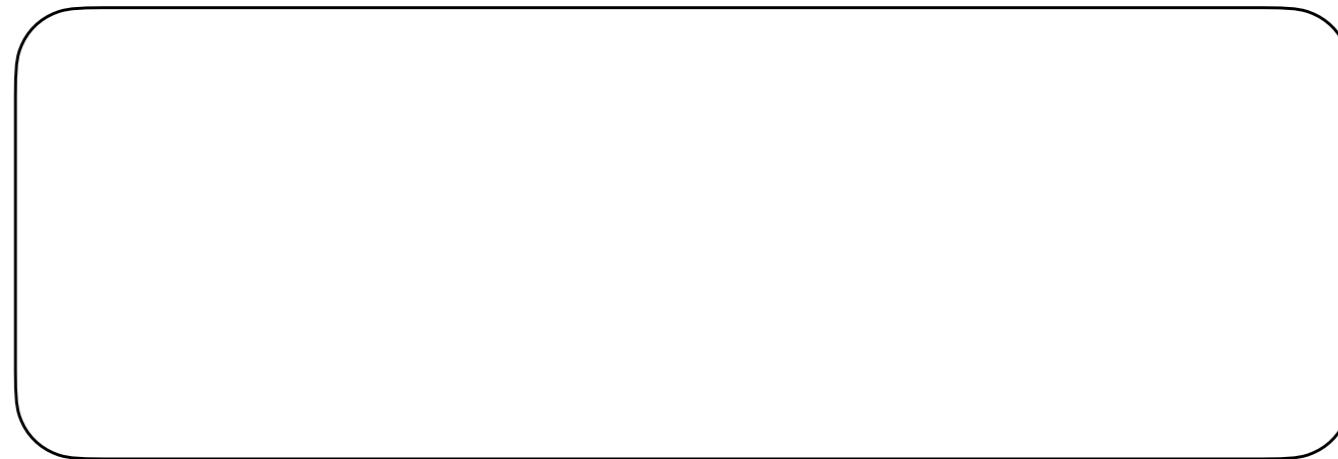
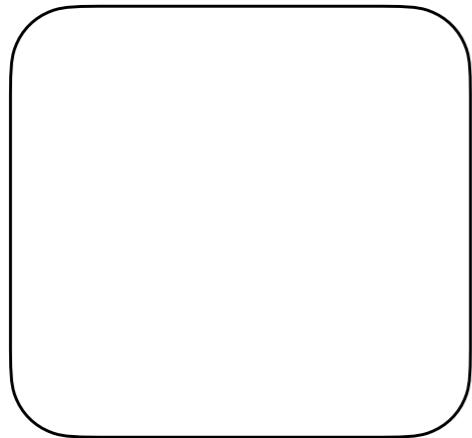
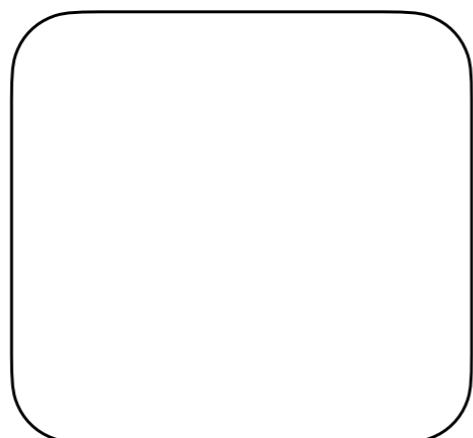
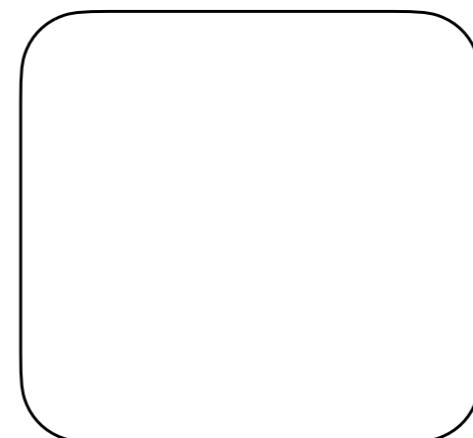
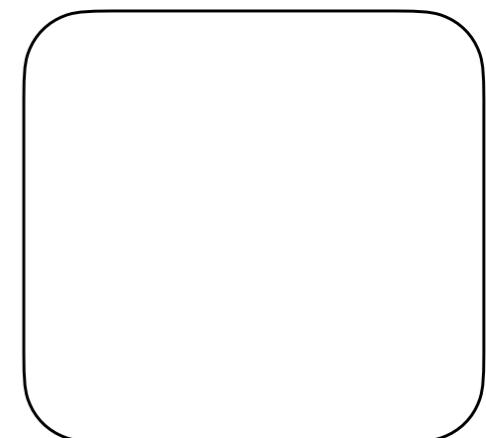
- During peak load there will be an maximum of 100000 active users connected to the portal
-

Architectural Design

- # Logical View (decomposition decisions)
- # Deployment View (deployment decisions)
- # Security View (Security decisions)
- # ...

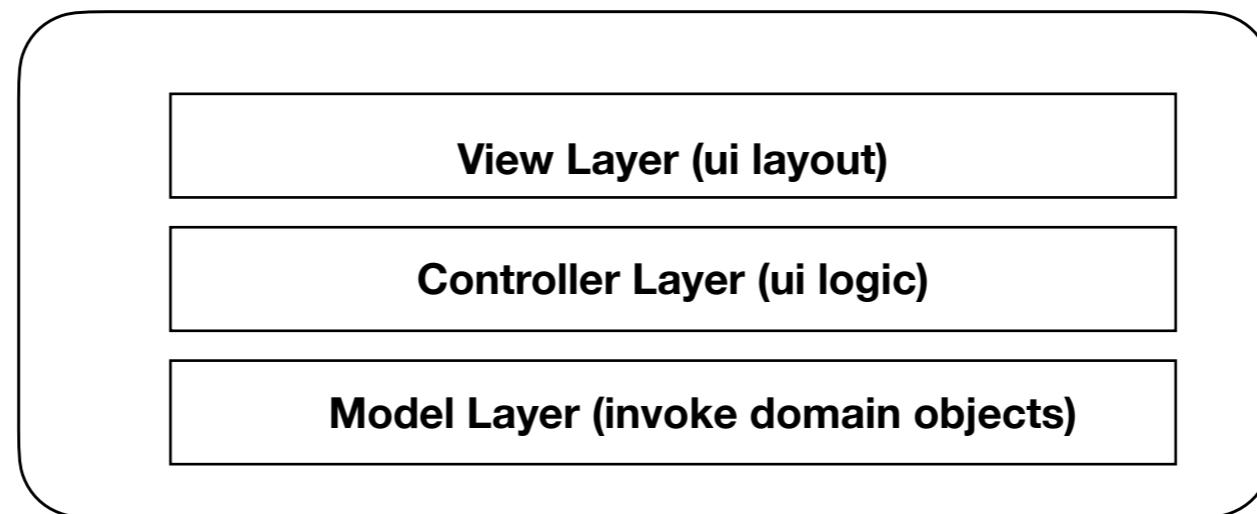
Logical View

- # System Decomposition**
- # Persistence approach**
- # Compute approach**
- # Communication approach**
- # cross cutting approach**

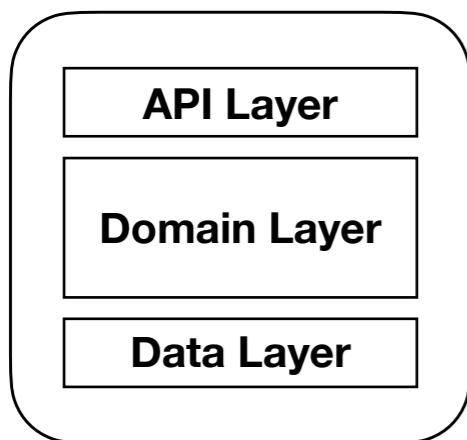
Web Portal**Listing Service****Booking Service****User Service****Search Service**

5 -> 1, 4

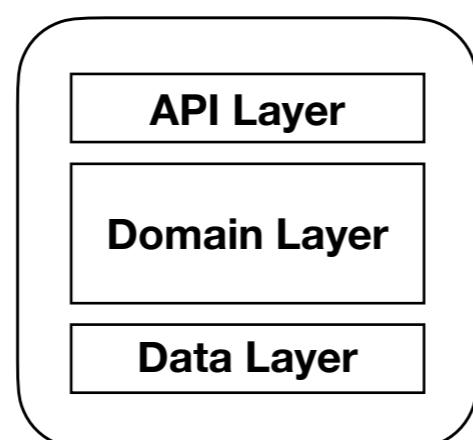
Web Portal



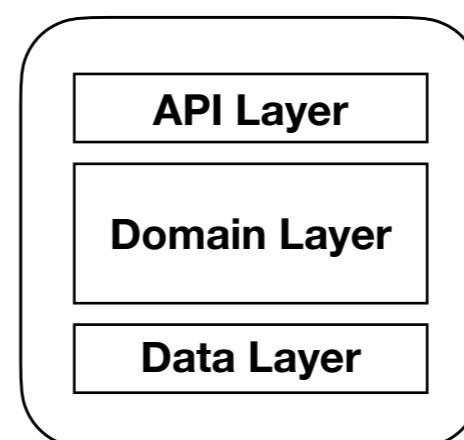
Listing Service



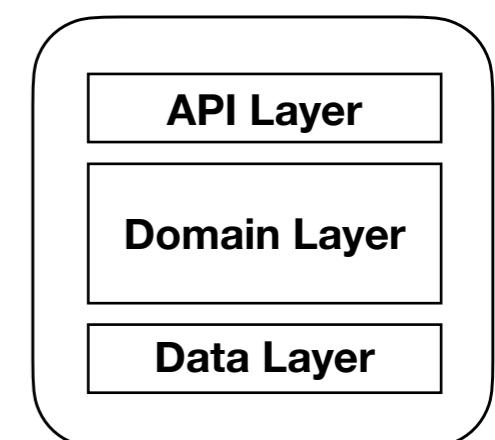
Booking Service



User Service

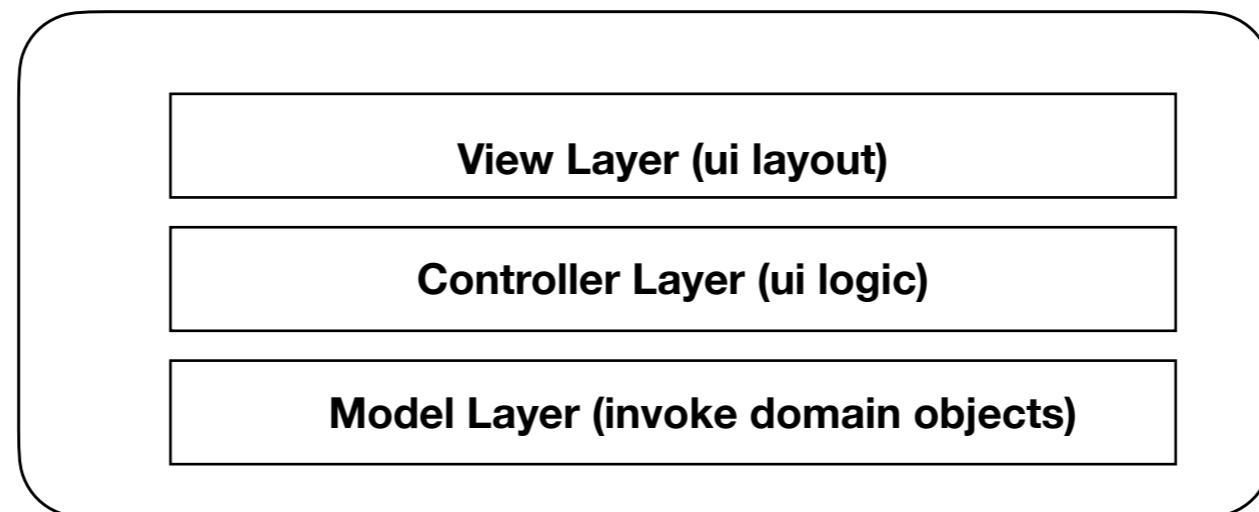


Search Service

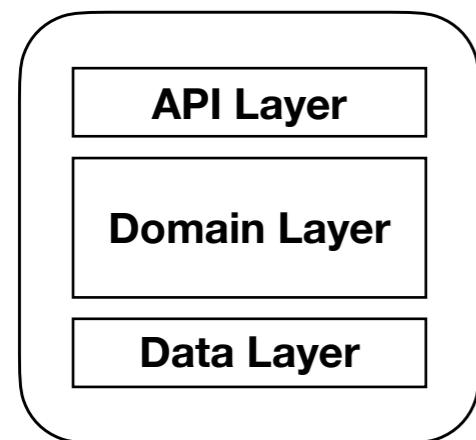


5 -> 1, 4

Web Portal



Listing Service

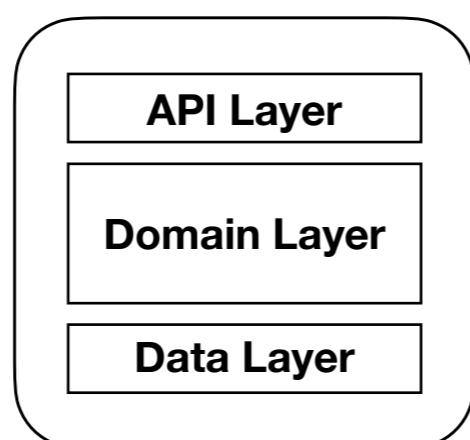


Object



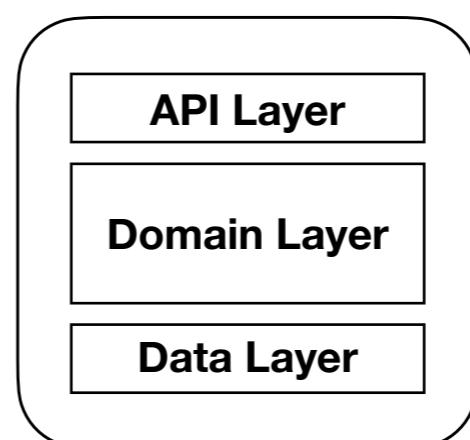
Document

Booking Service



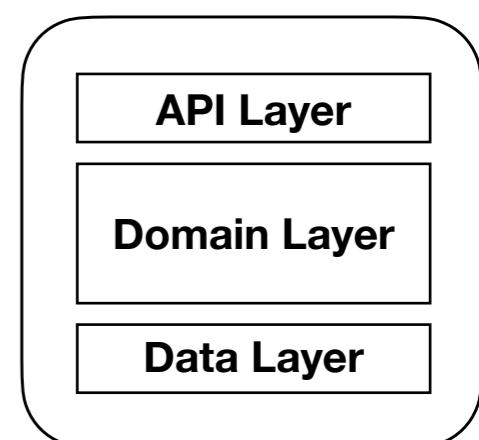
Document

User Service



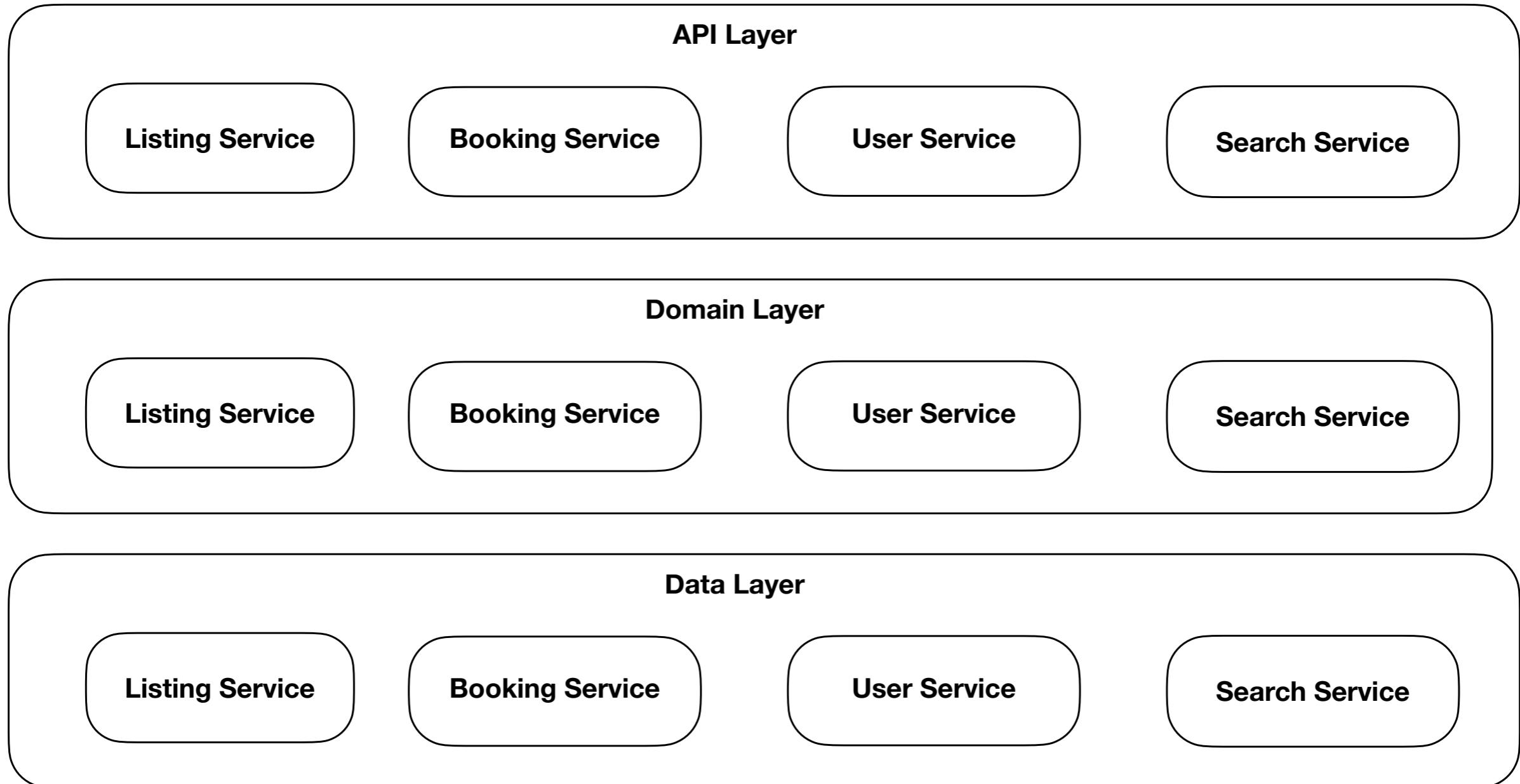
Document

Search Service

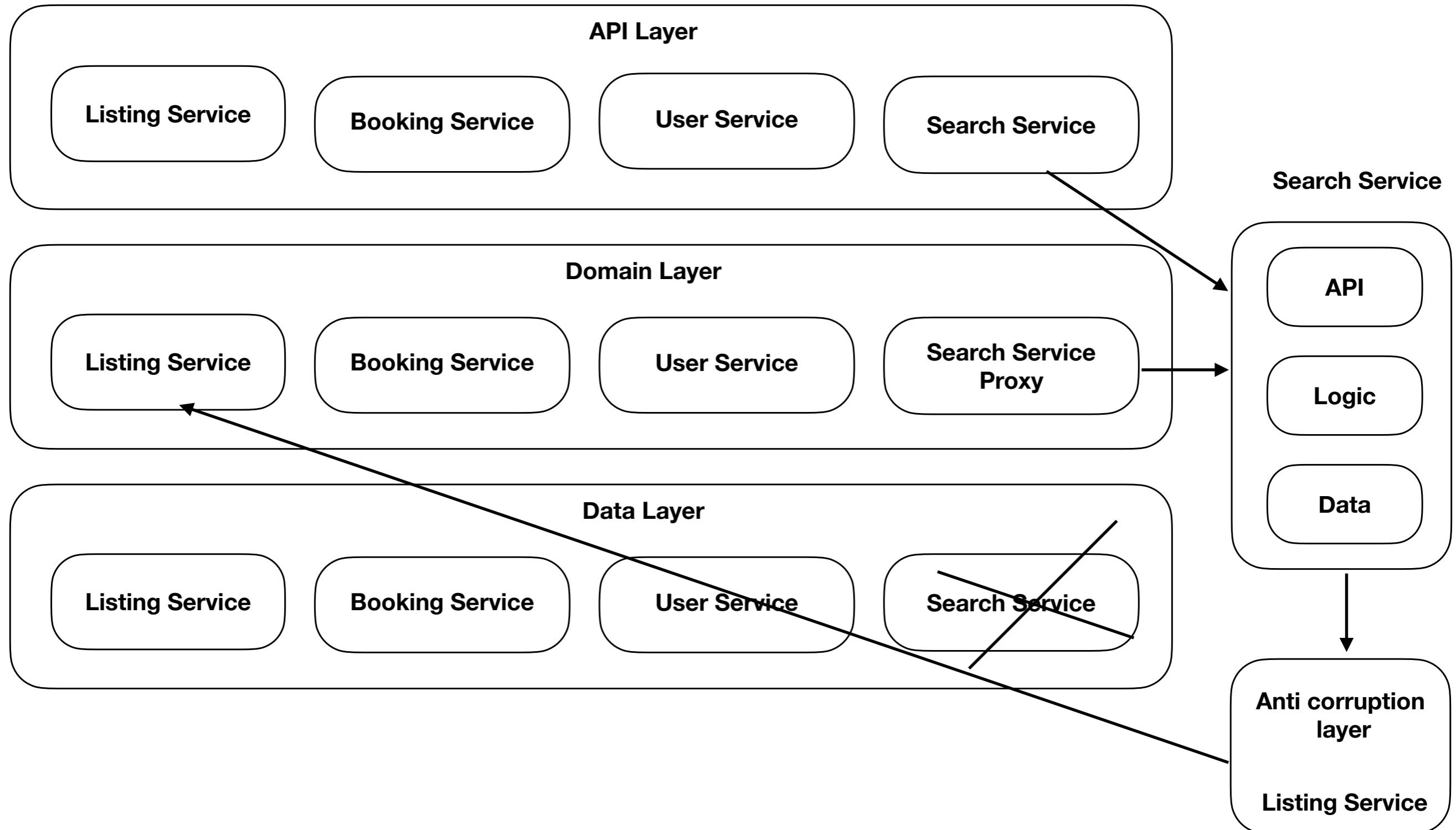


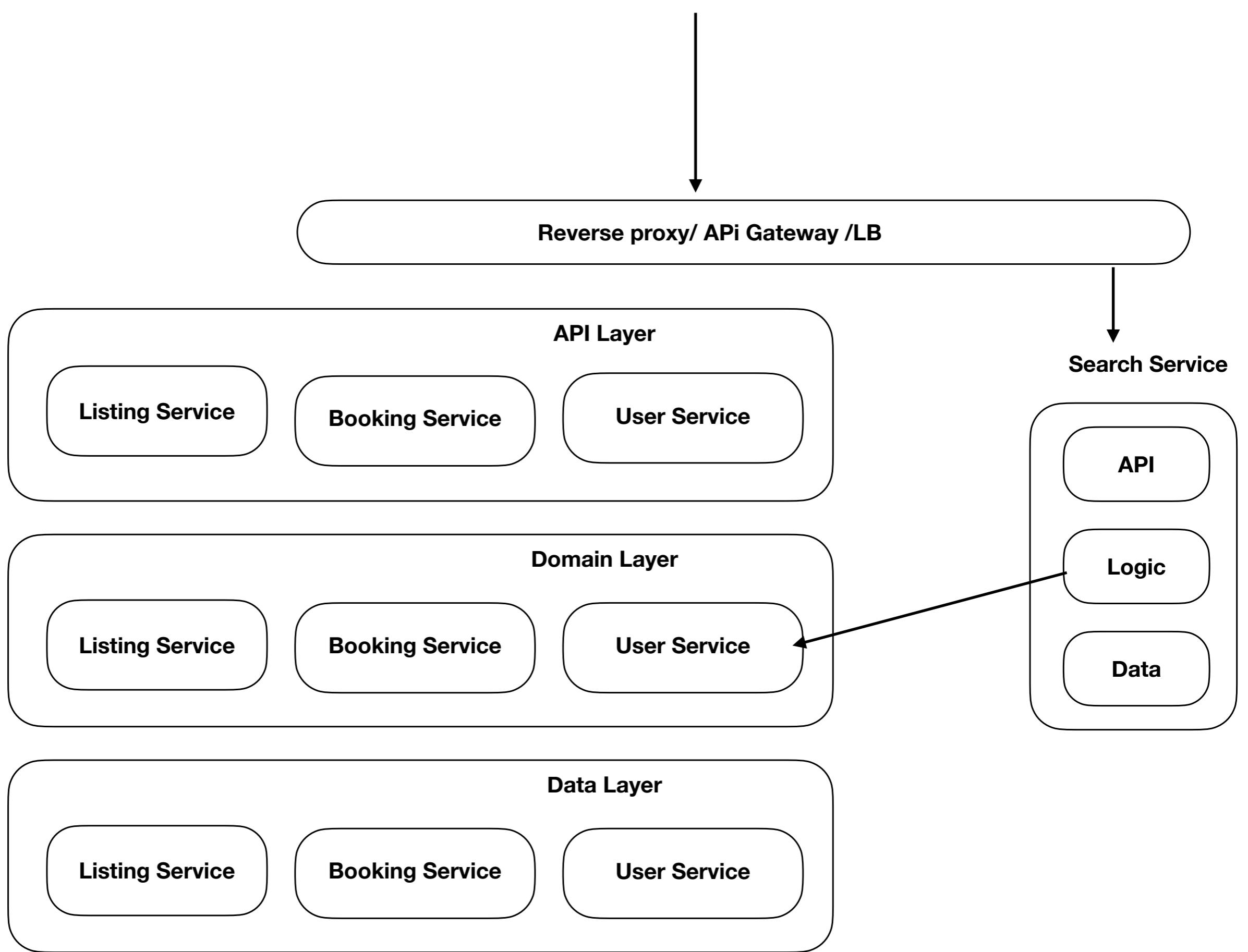
Text

1->5

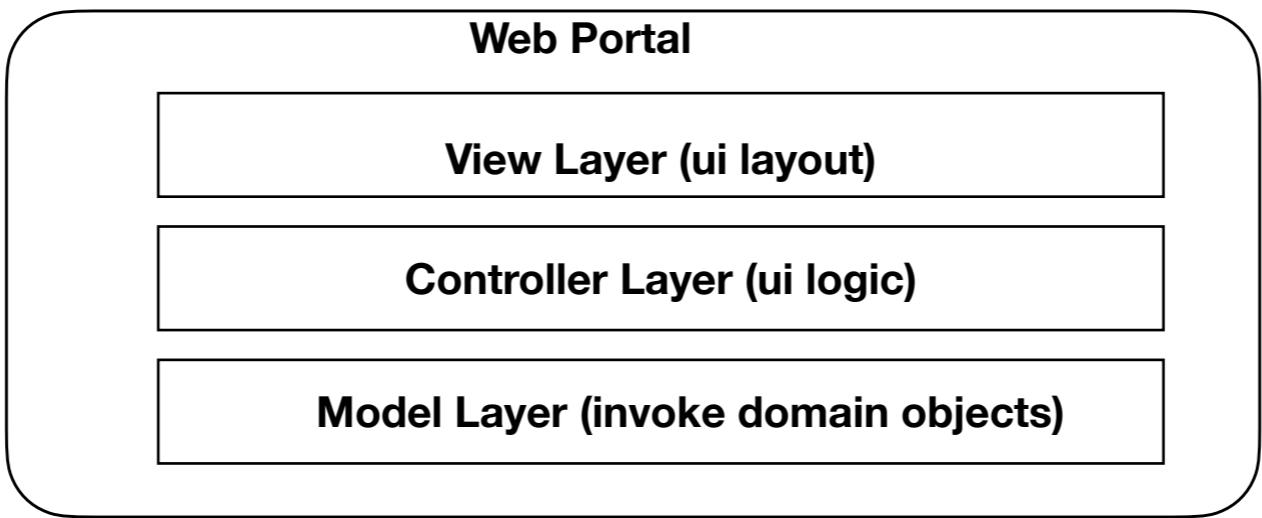


1->5

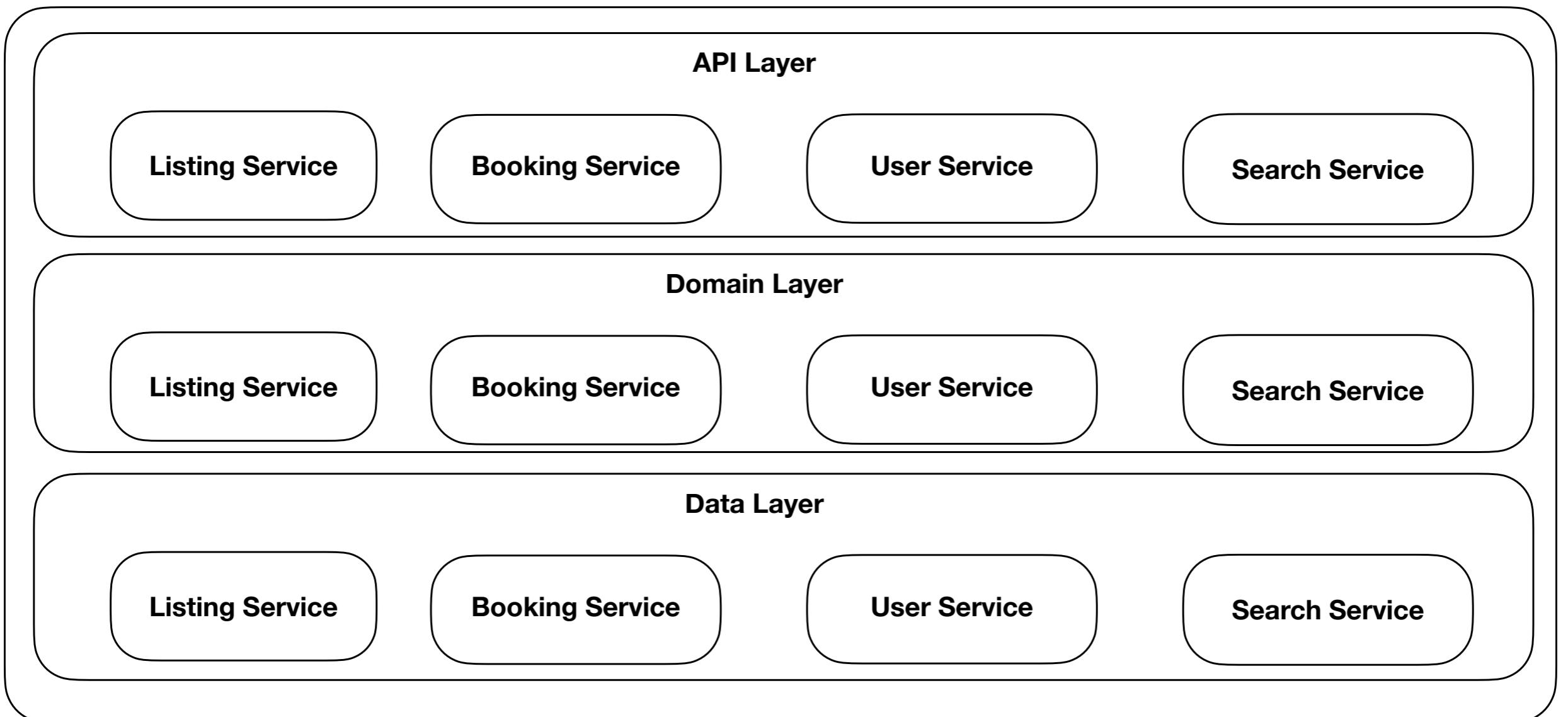




1->5, 4



T1



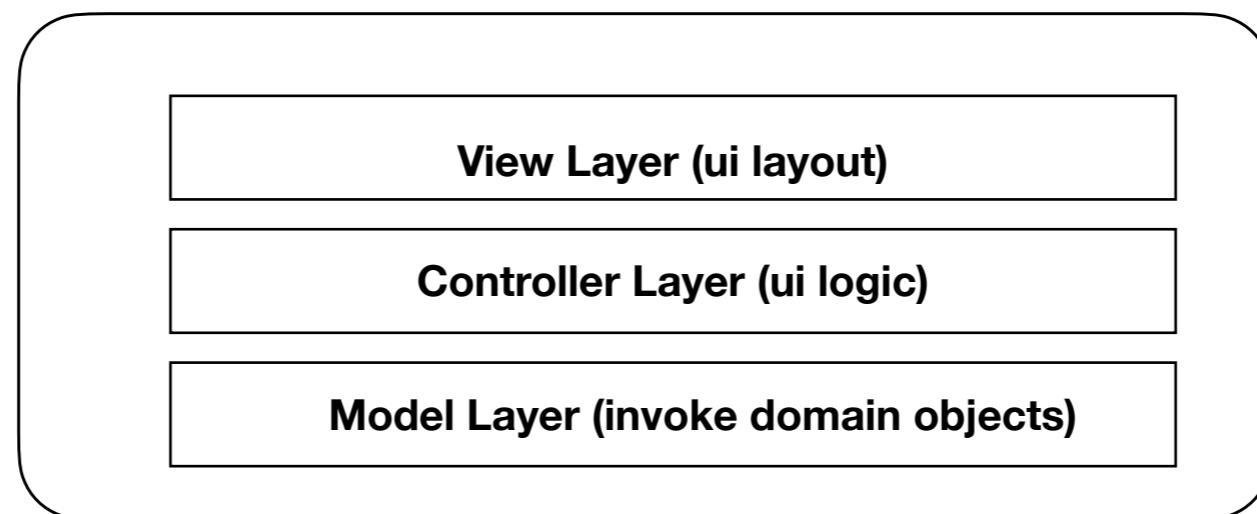
T2



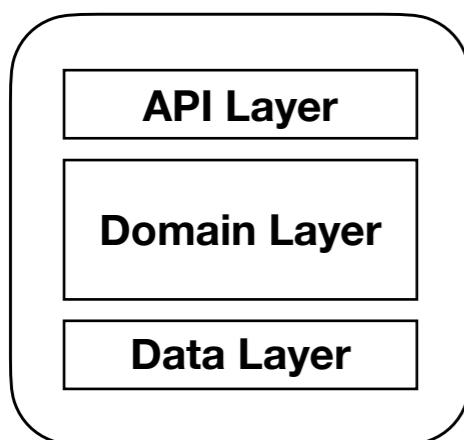
T3

5 -> 1, 4

Web Portal



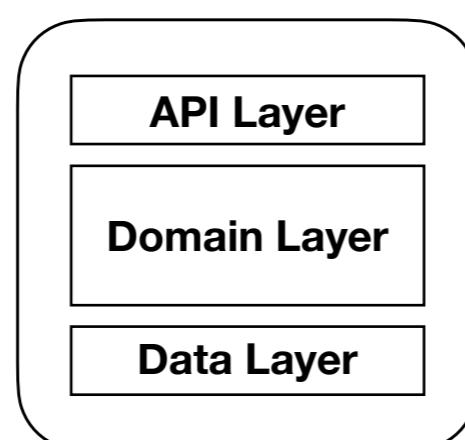
Listing Service



Doc db

flexi schema
multiple keys

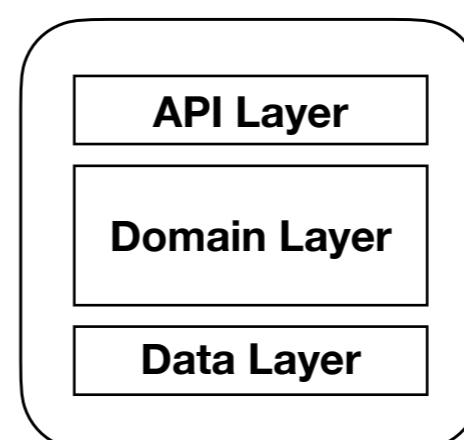
Booking Service



Rdbms or Doc db

strict schema
multiple keys
ACID

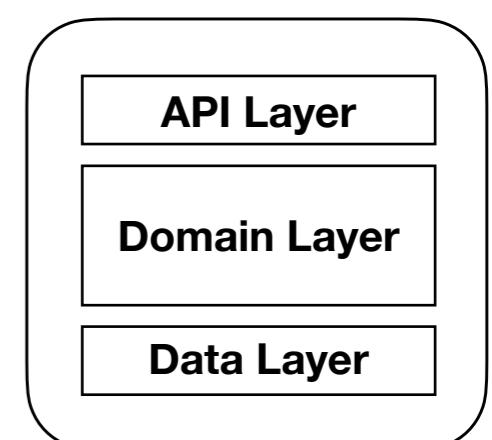
User Service



Rdbms or Doc db

strict schema
multiple keys

Search Service

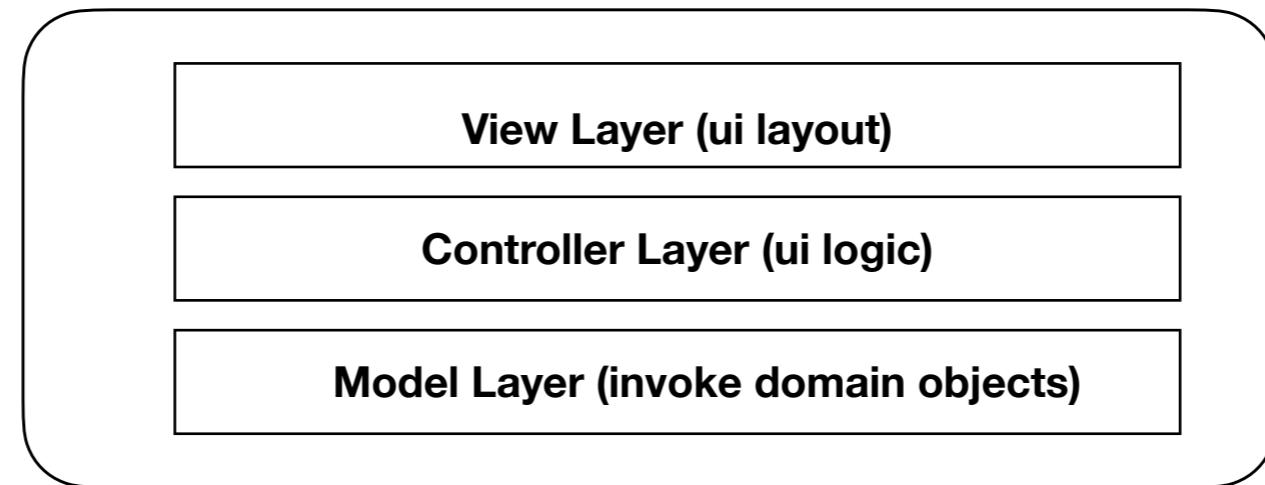


Text db

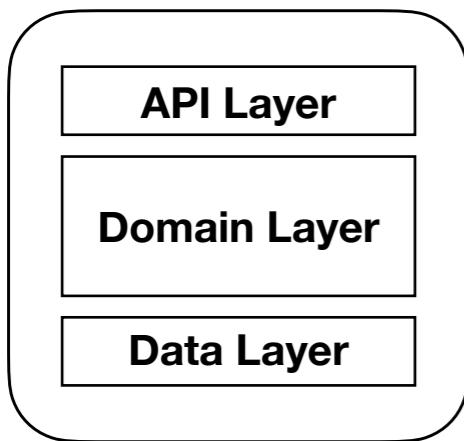
search

5 -> 1, 4

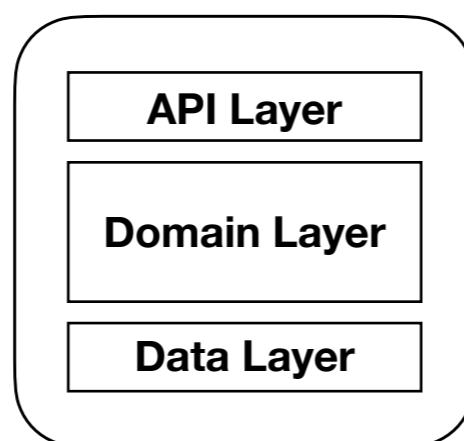
Web Portal



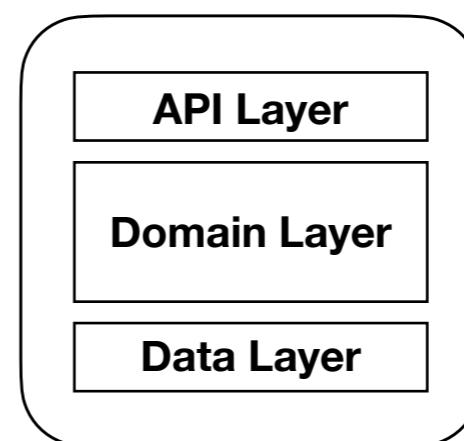
Listing Service



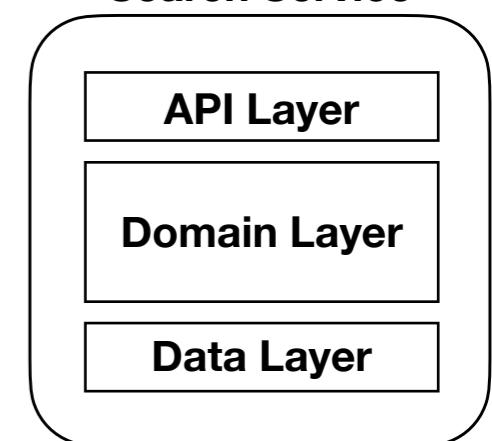
Booking Service



User Service



Listing Search Service



Current Rdbms Storage



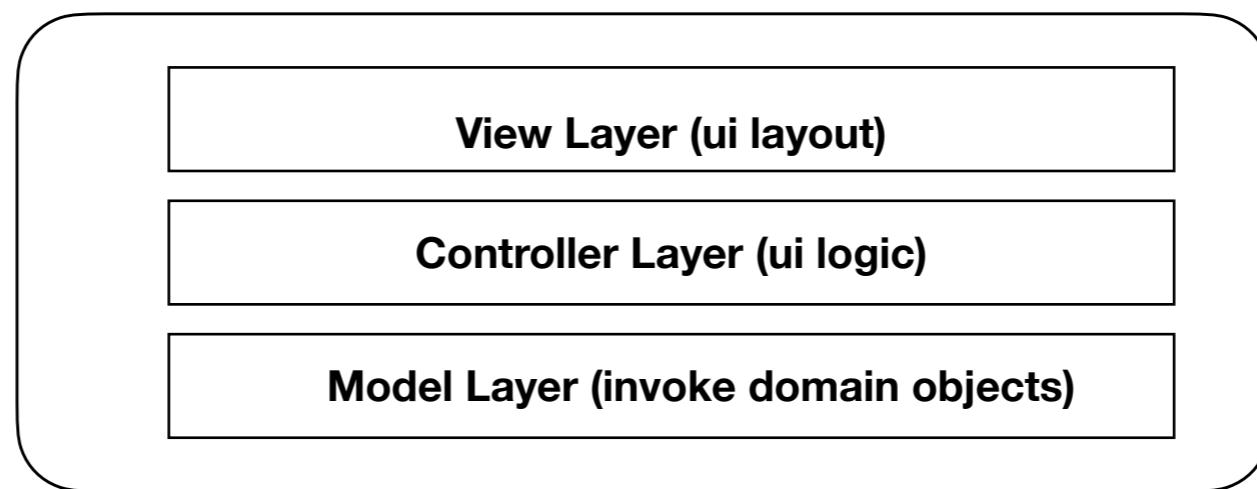
Rdbms Storage



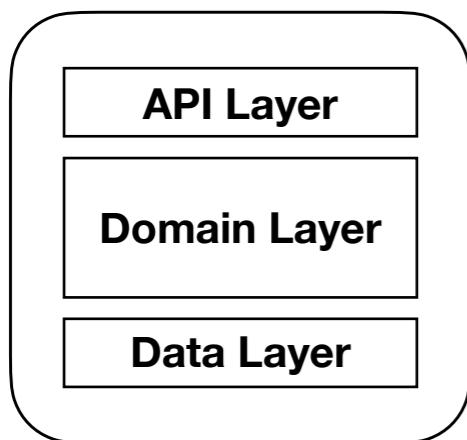
Text db

5 -> 1, 4

Web Portal



Listing Service

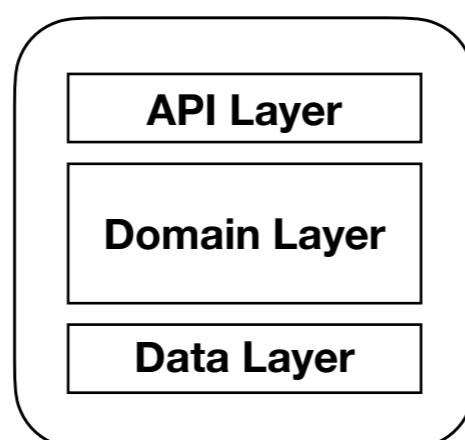


Object storage



Document Storage

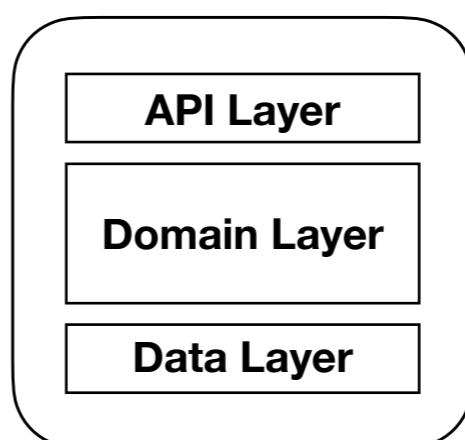
Booking Service



Current

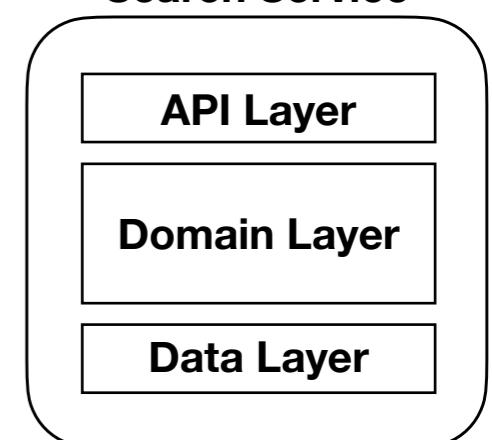
Document / Rdbms Storage

User Service



Document / Rdbms Storage

Listing Search Service

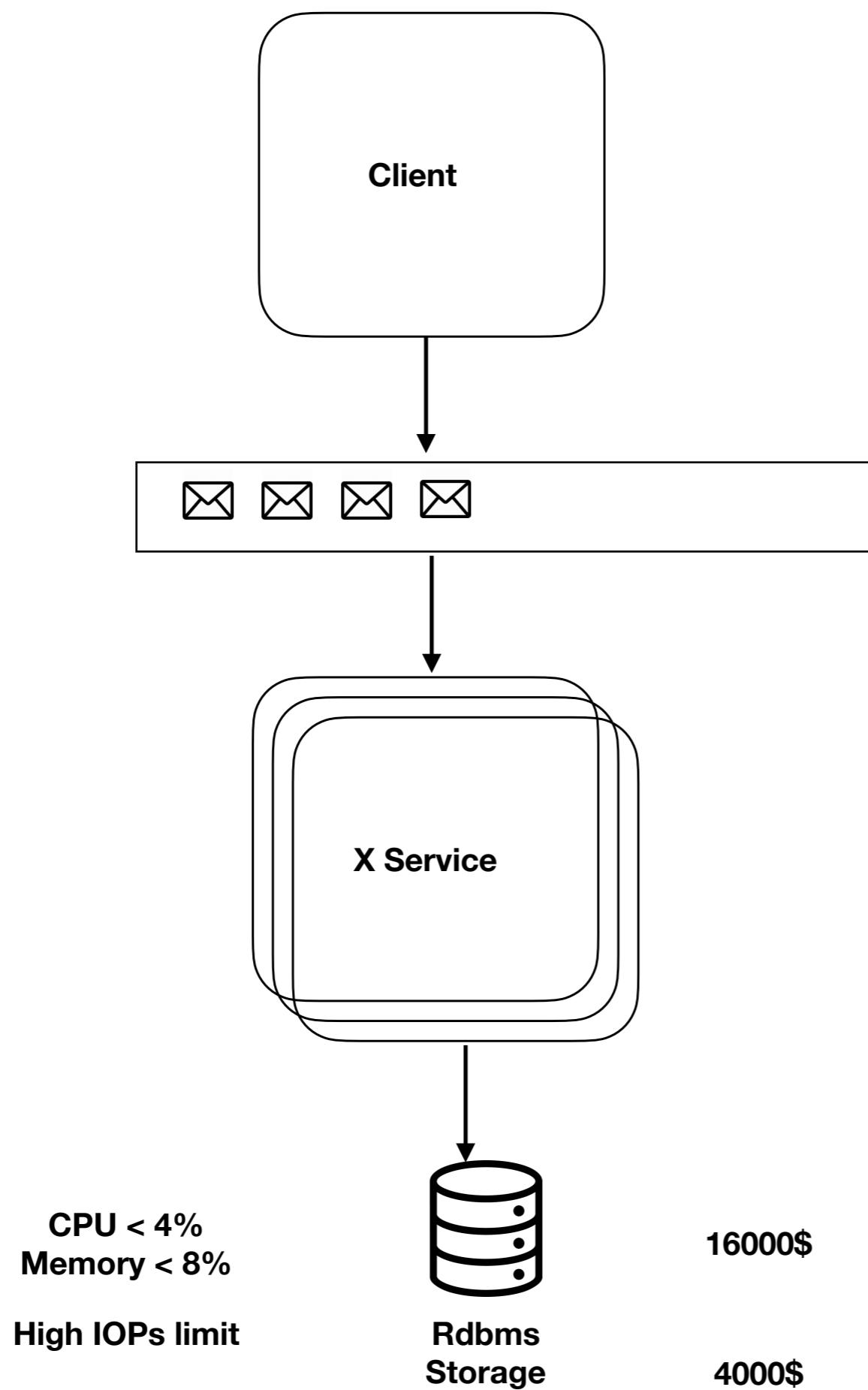


Text db

View Layer (ui layout)

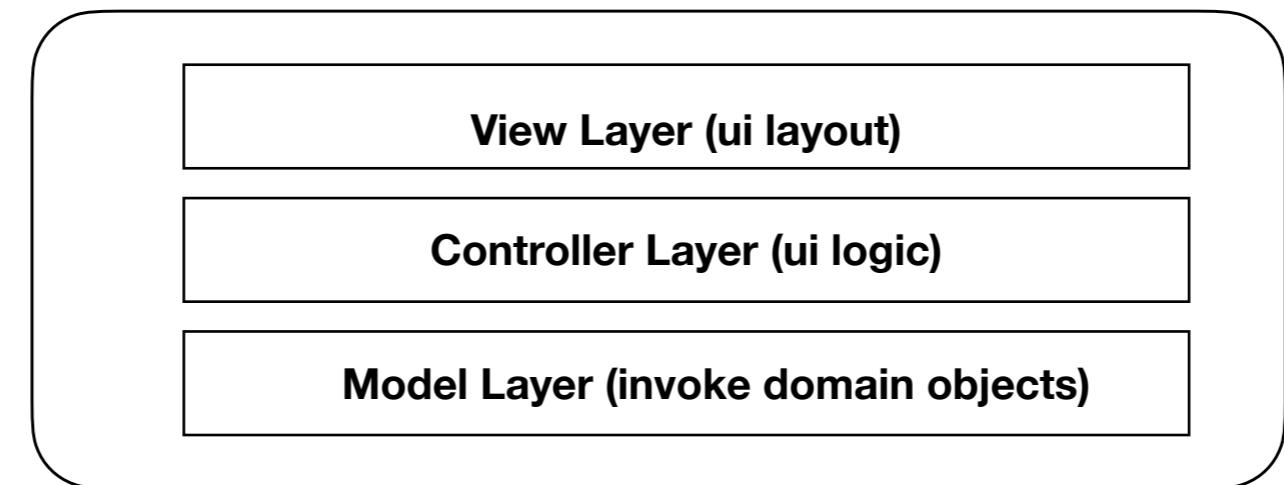
Controller Layer (ui logic)

Model Layer (invoke domain objects)

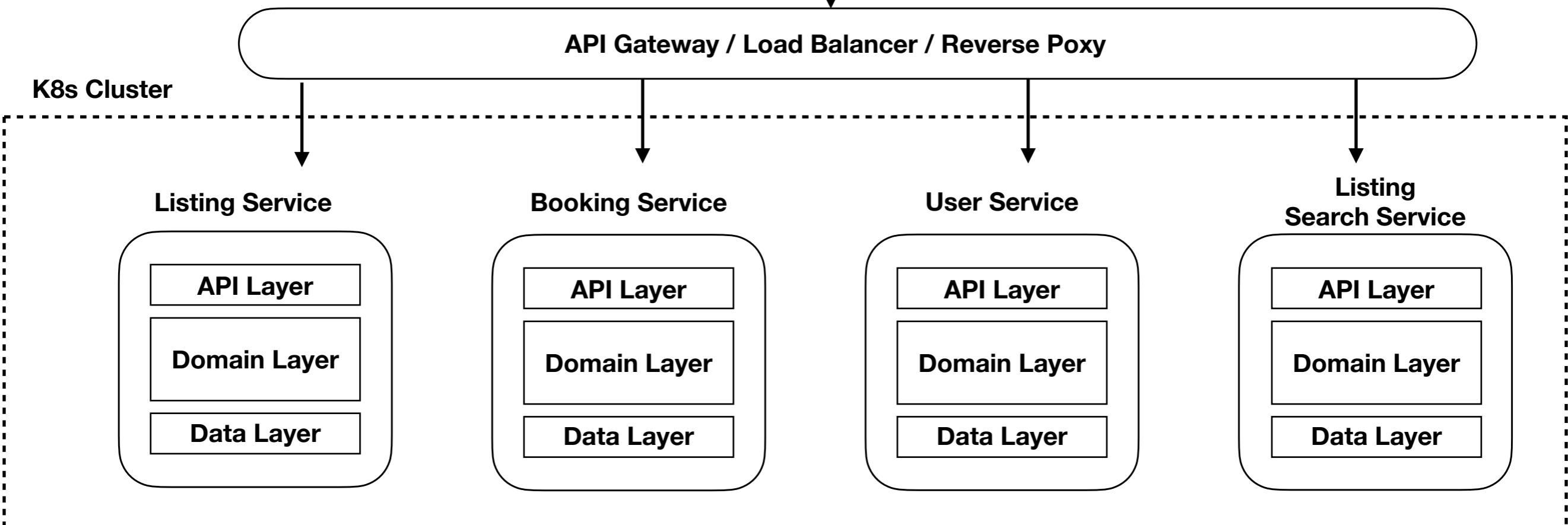


Web Portal

5 -> 1, 4



K8s Cluster



Object
storage



Document
Storage



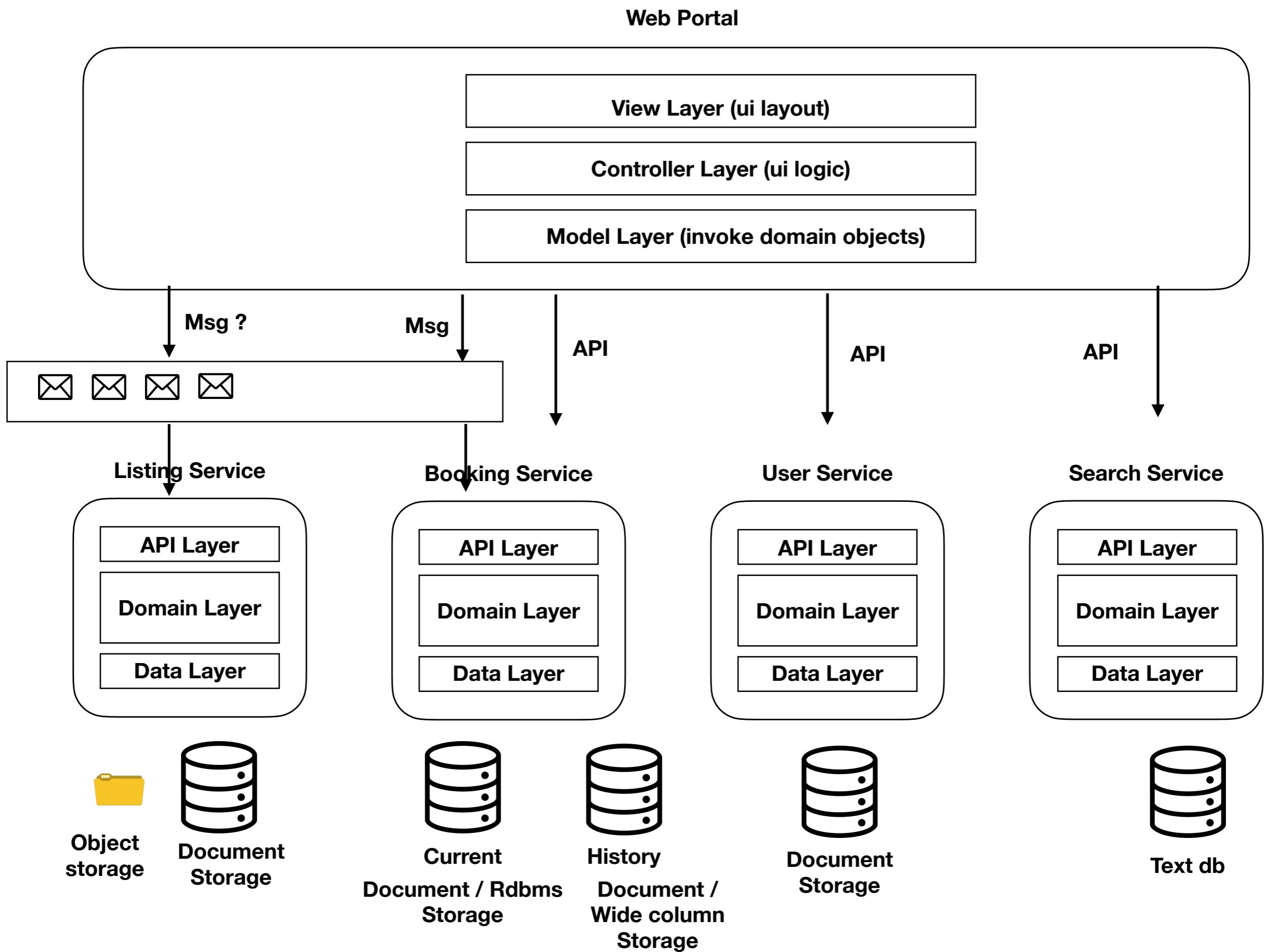
Current
Rdbms
Storage

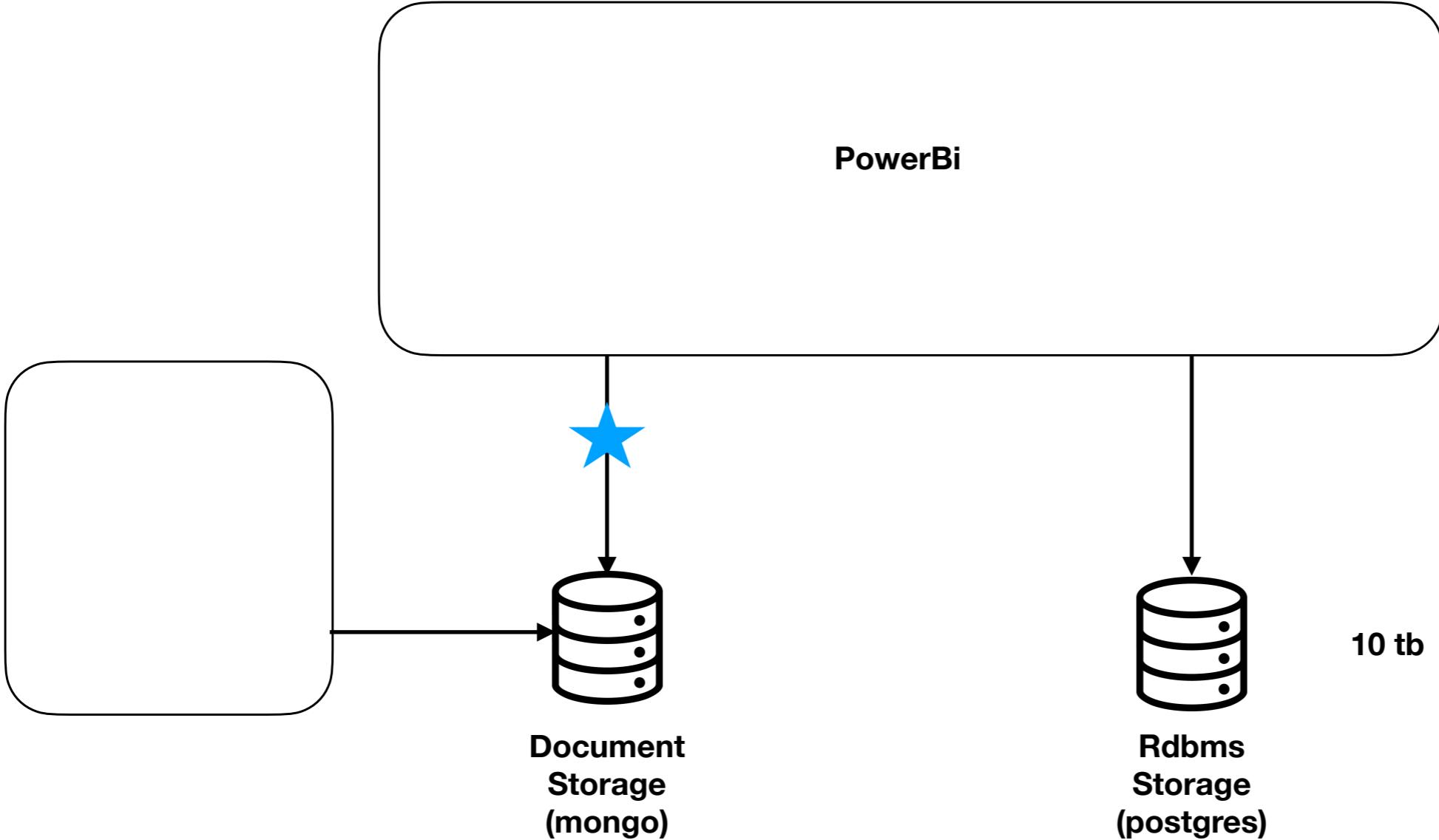


Rdbms
Storage



Text db

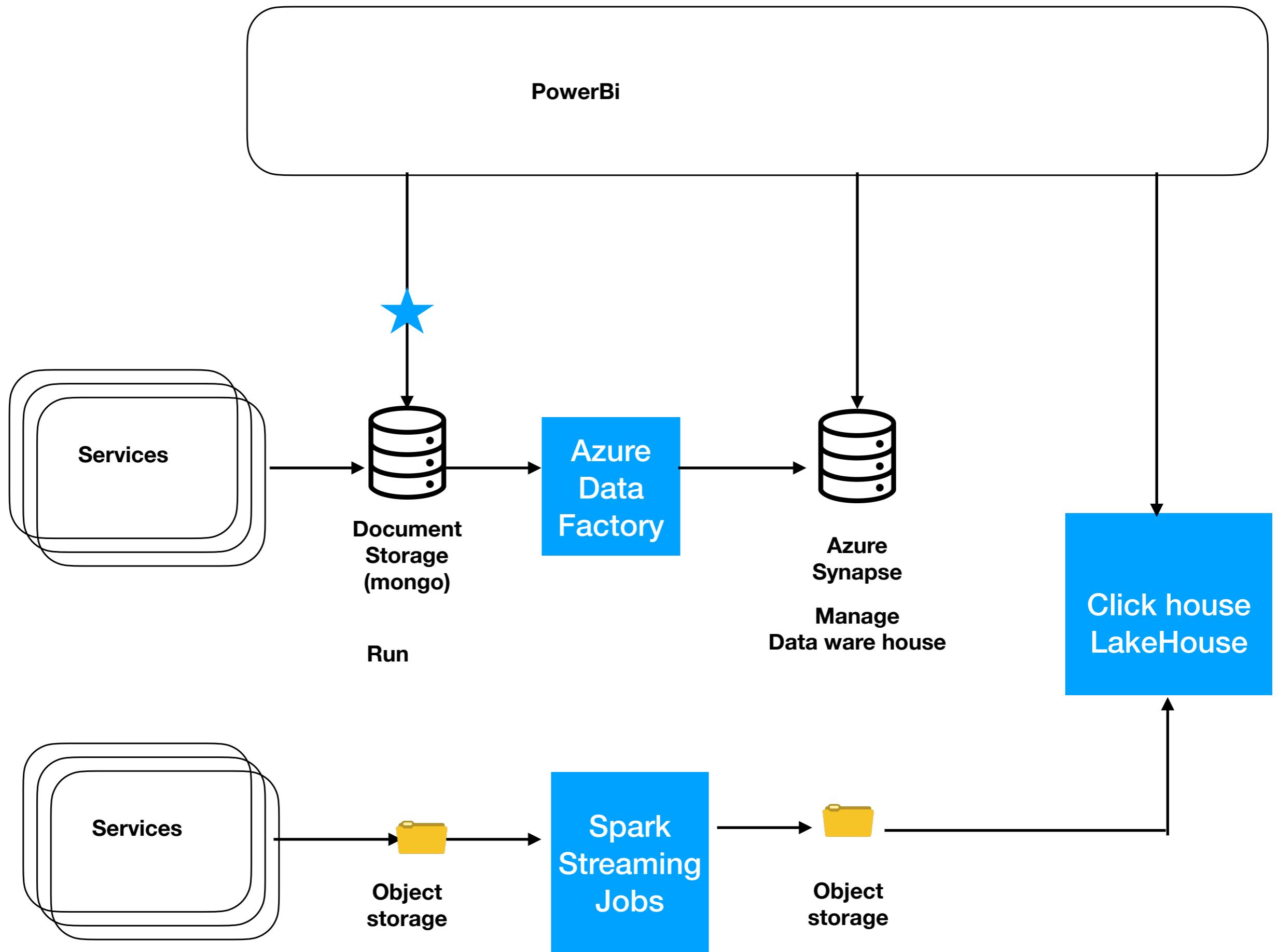


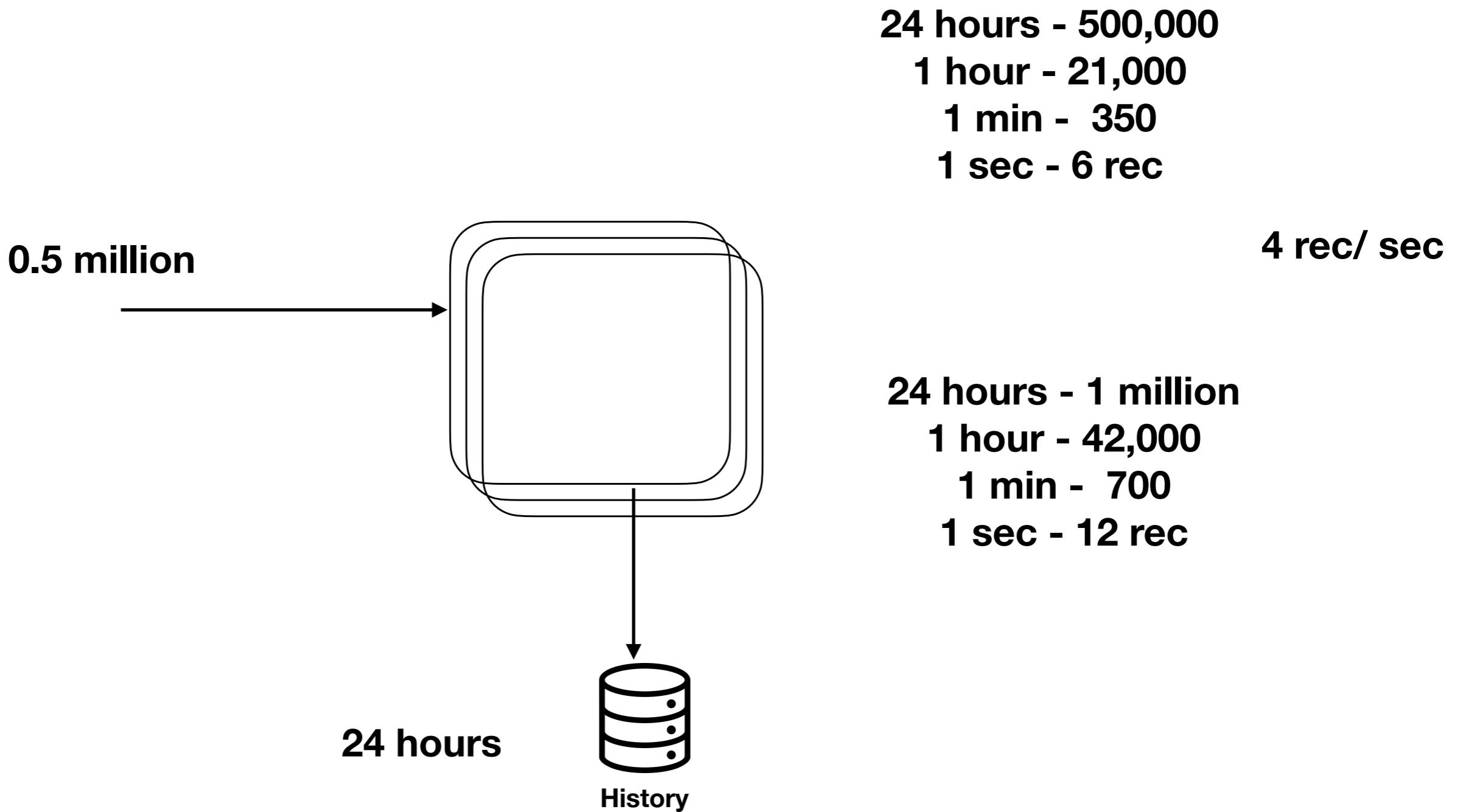


Schema was Read Optimized
- denormalized

Emp (emp_id, dept_id,dept_name)

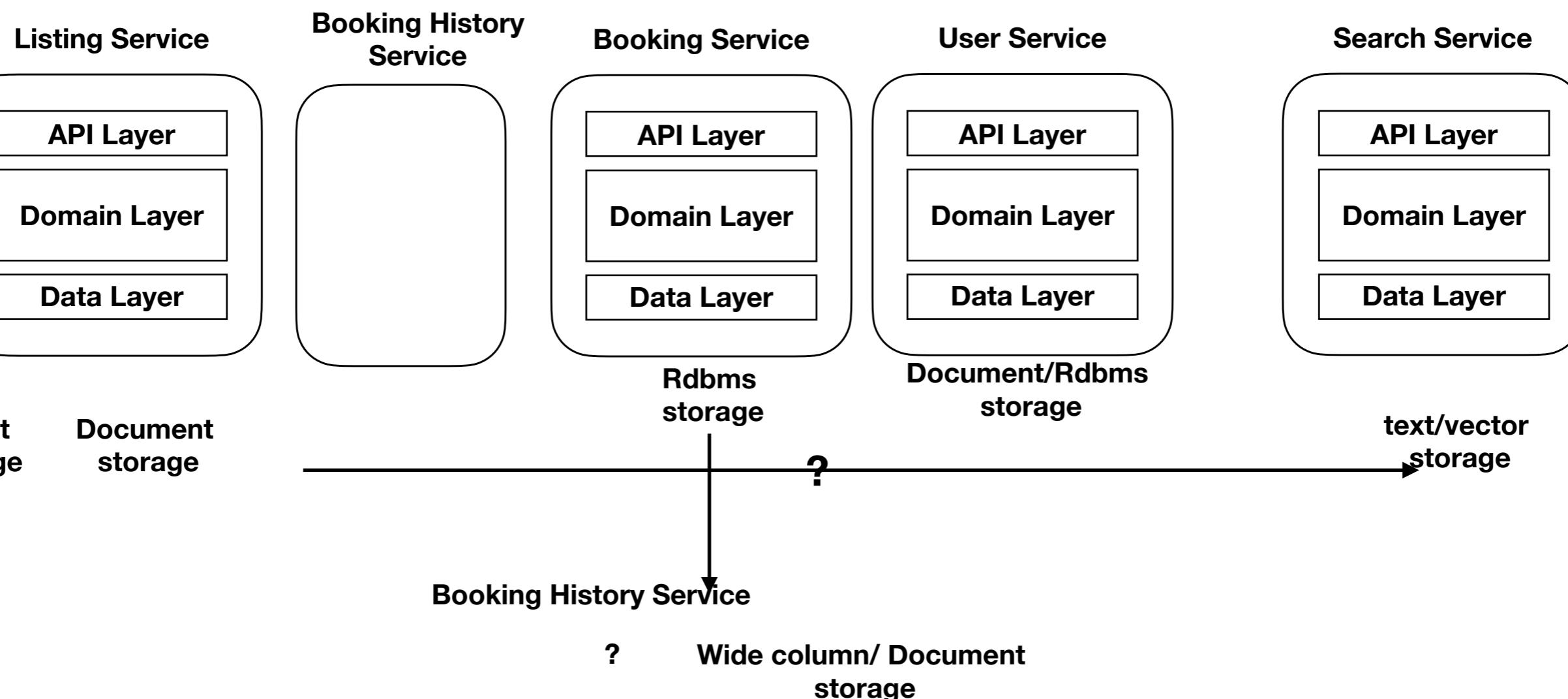
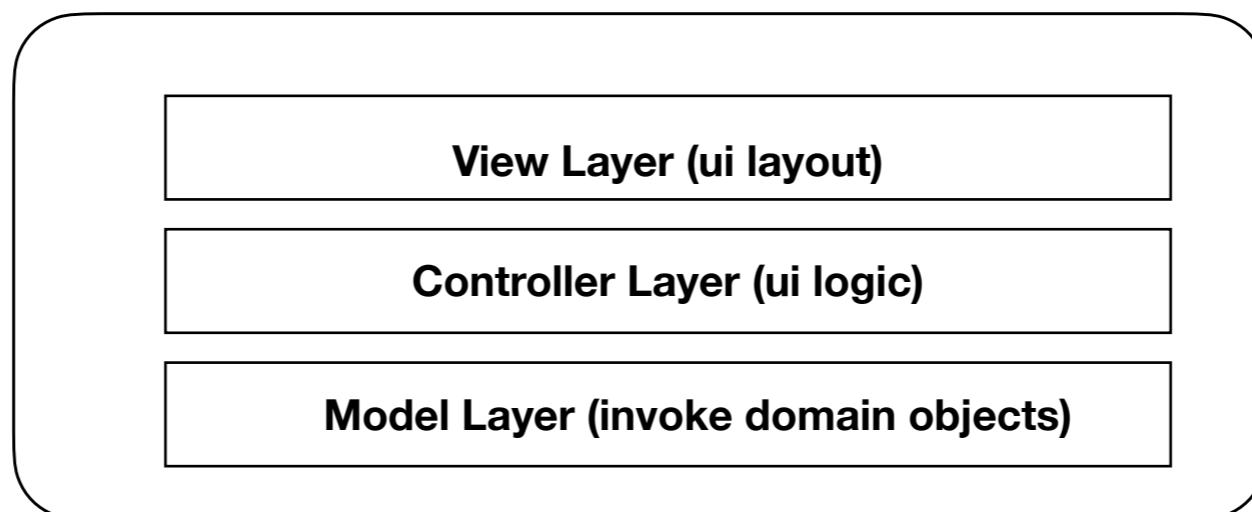
Dept (id, name)





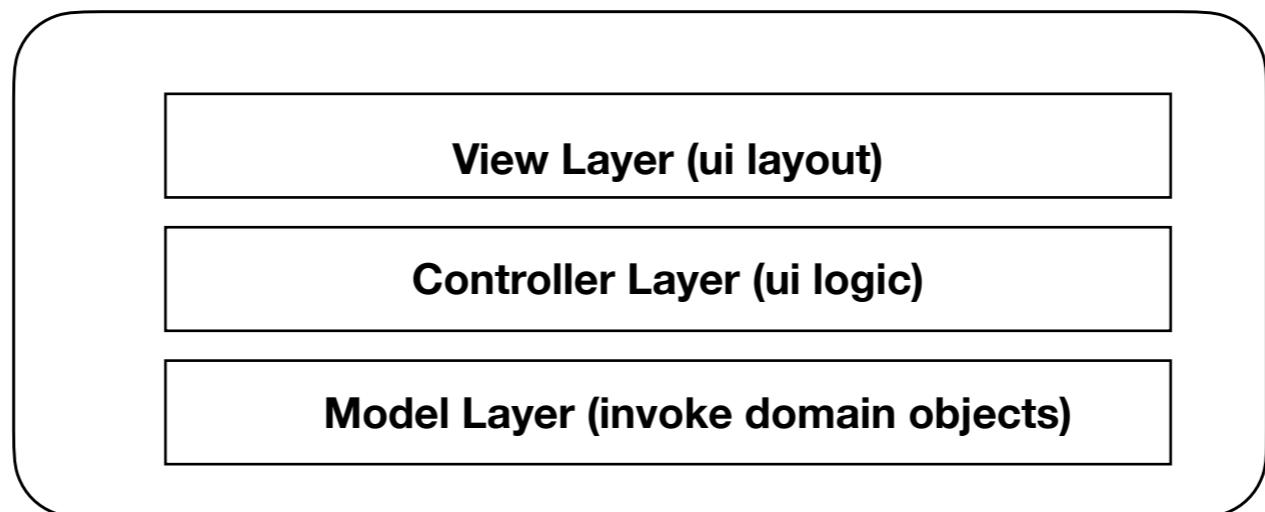
5 -> 1, 4

Web Portal

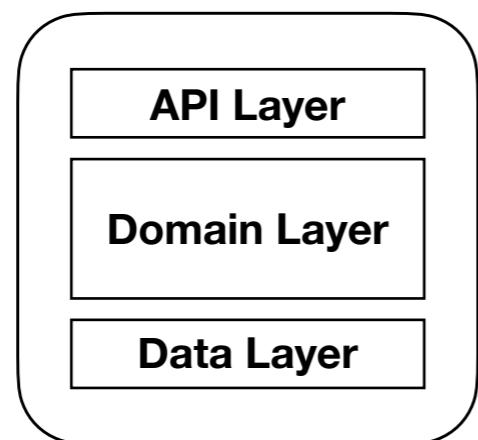


5 -> 1, 4

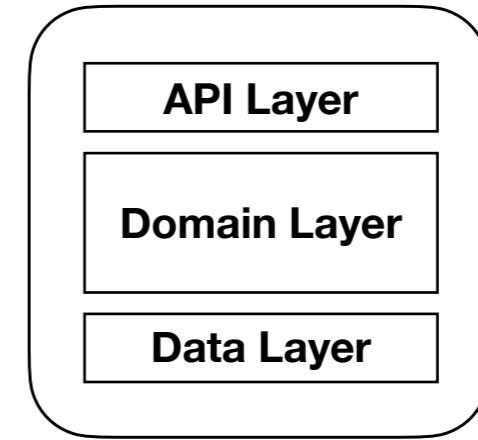
Web Portal



Listing Service



Search Service

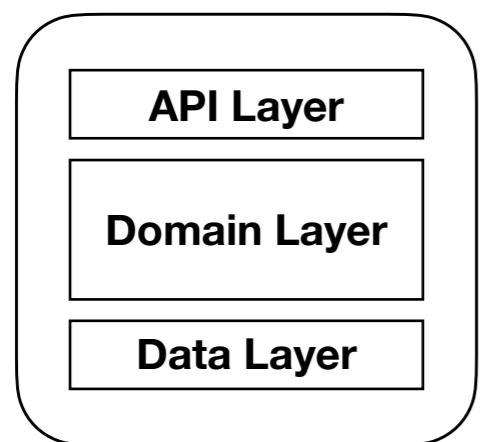


Document
storage

?

text/vector
storage

Listing Service

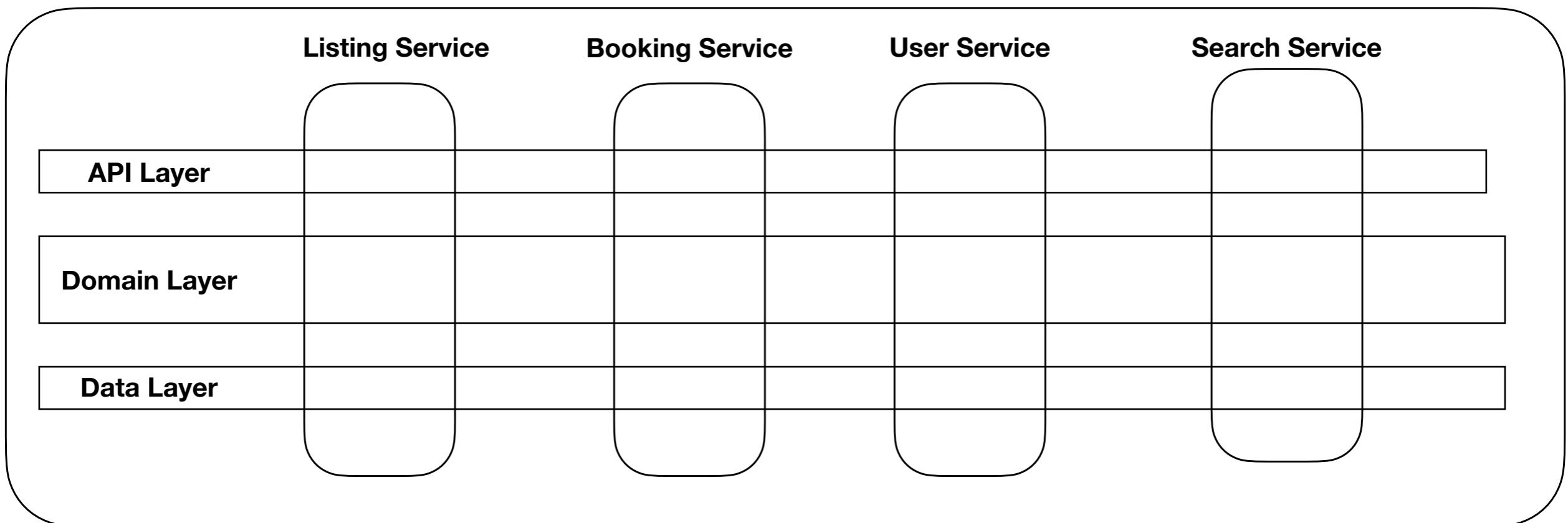
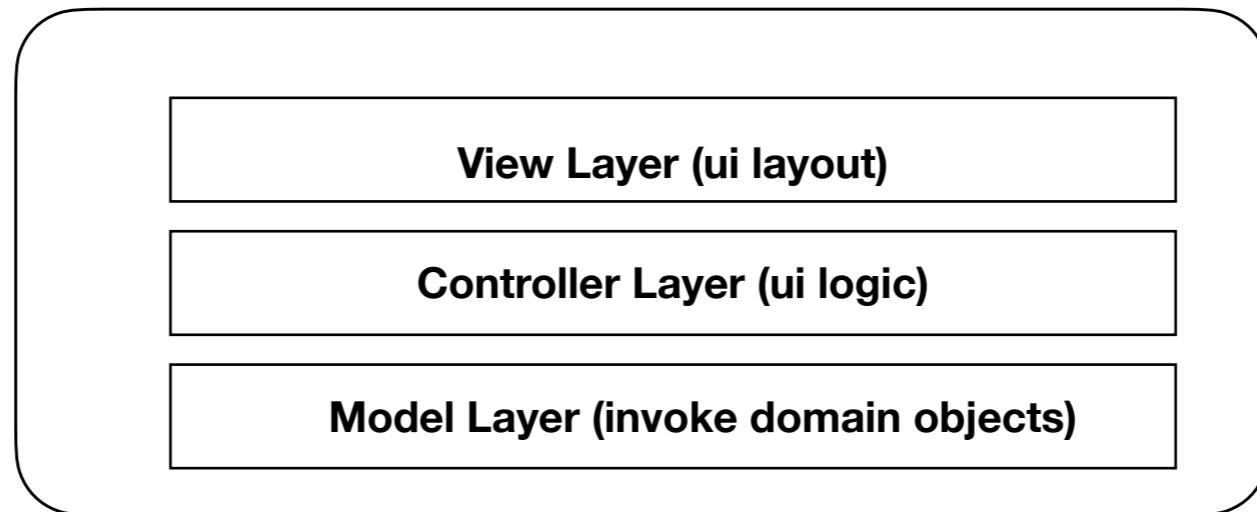


**Object Document
storage storage**

4

1 -> 5

Web Portal

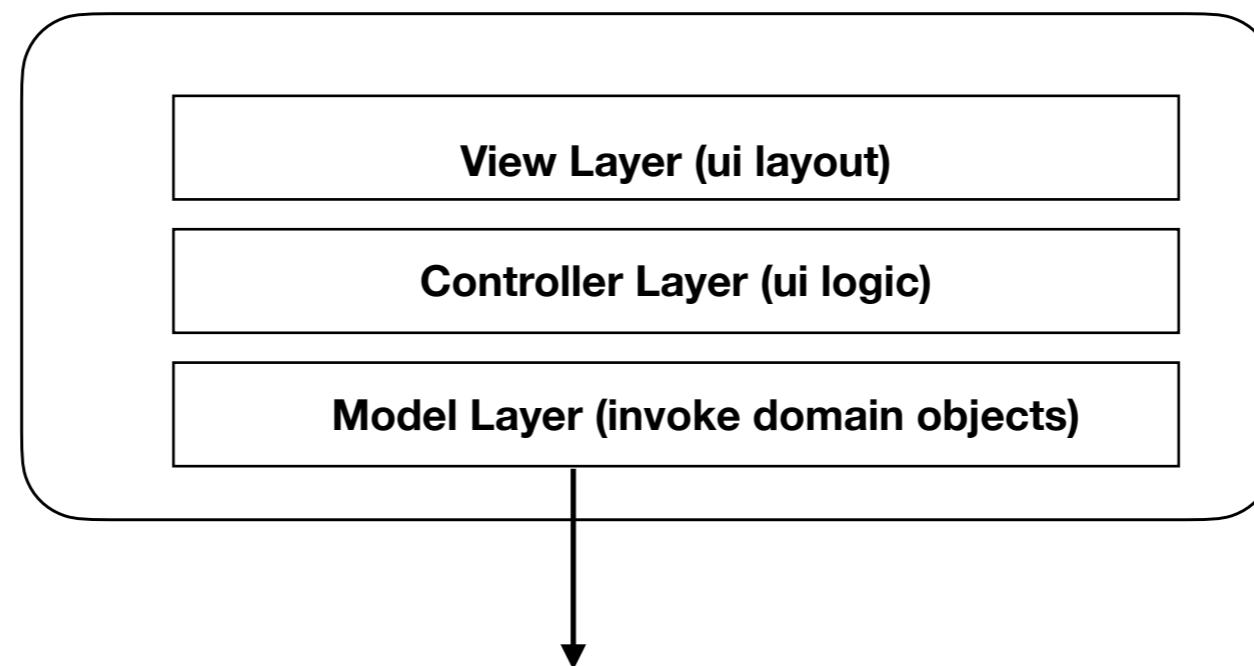


Microservice

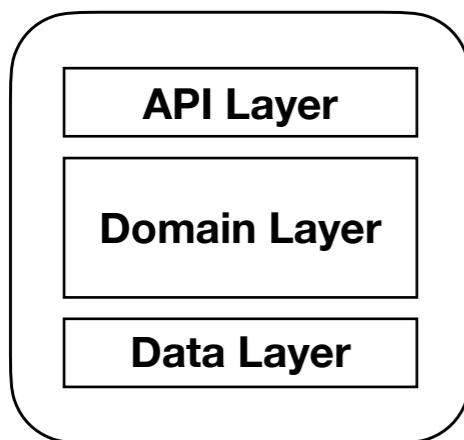
MVC

Layered

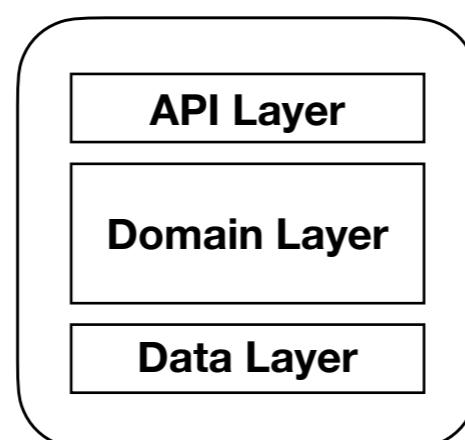
Web Portal



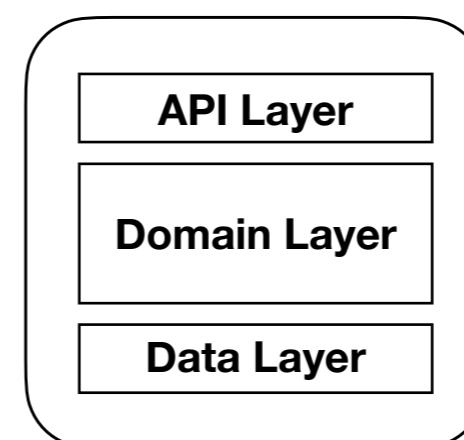
Listing Service



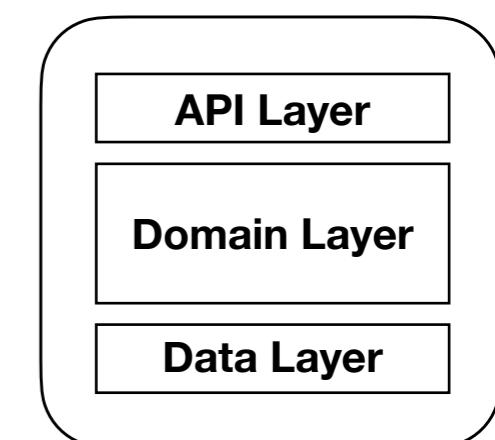
Booking Service



User Service

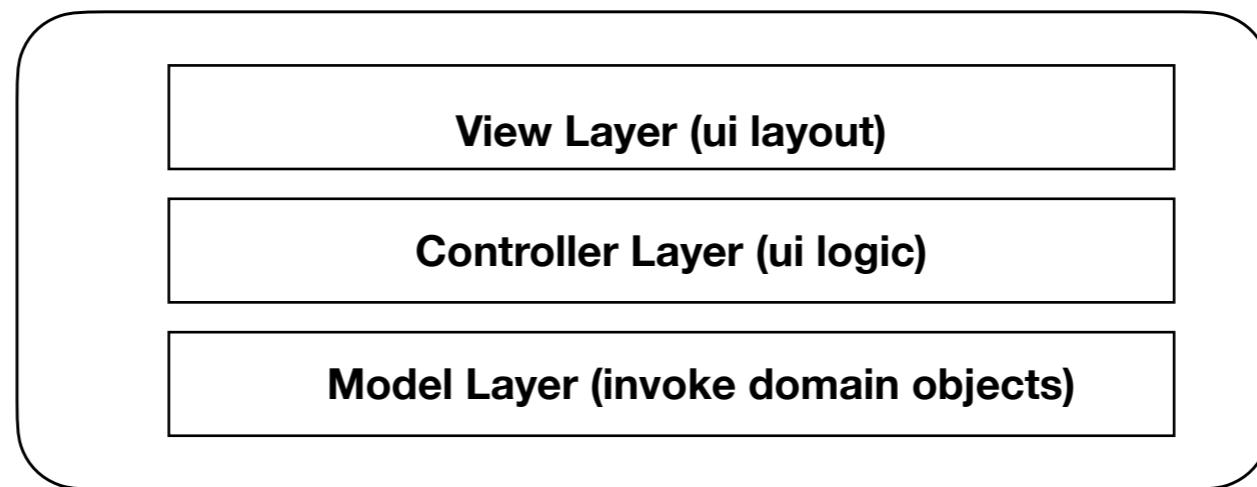


Search Service

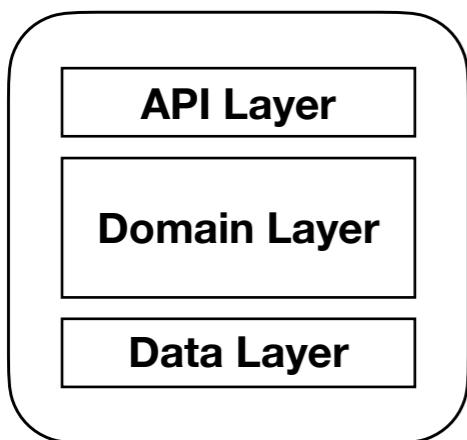


5 -> 1, 4

Web Portal



Listing Service

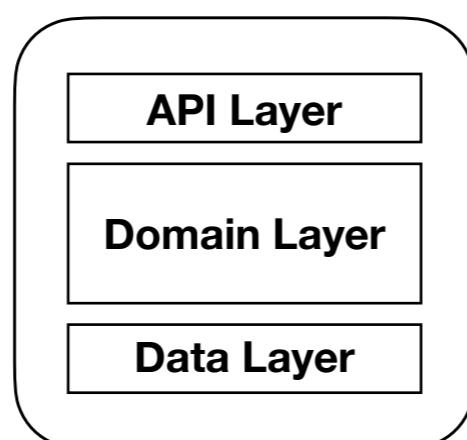


Object storage



Document Storage

Booking Service

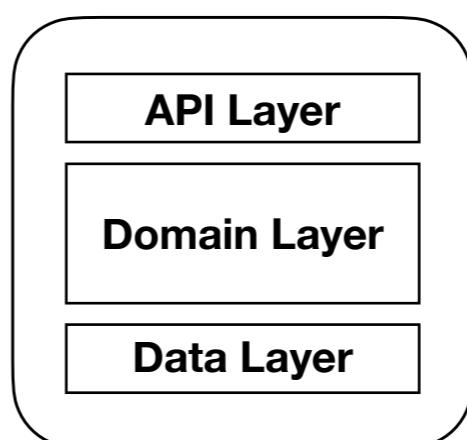


Current Document / Rdbms Storage



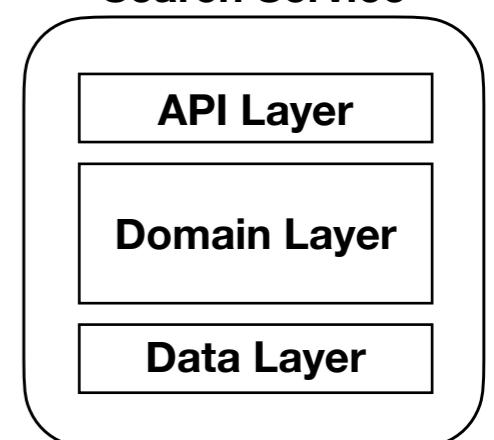
History Document / * Wide column Storage

User Service



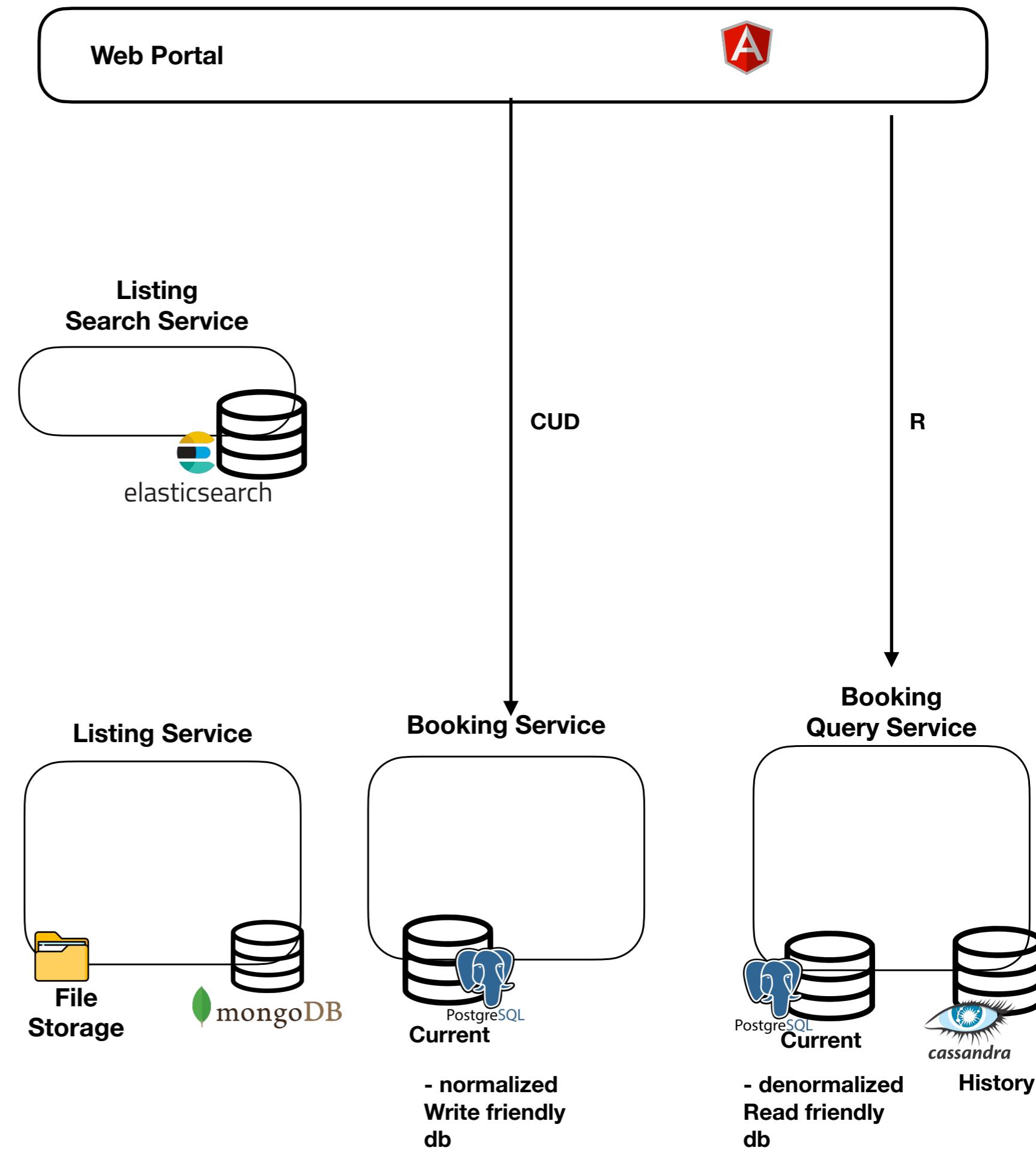
Document Storage

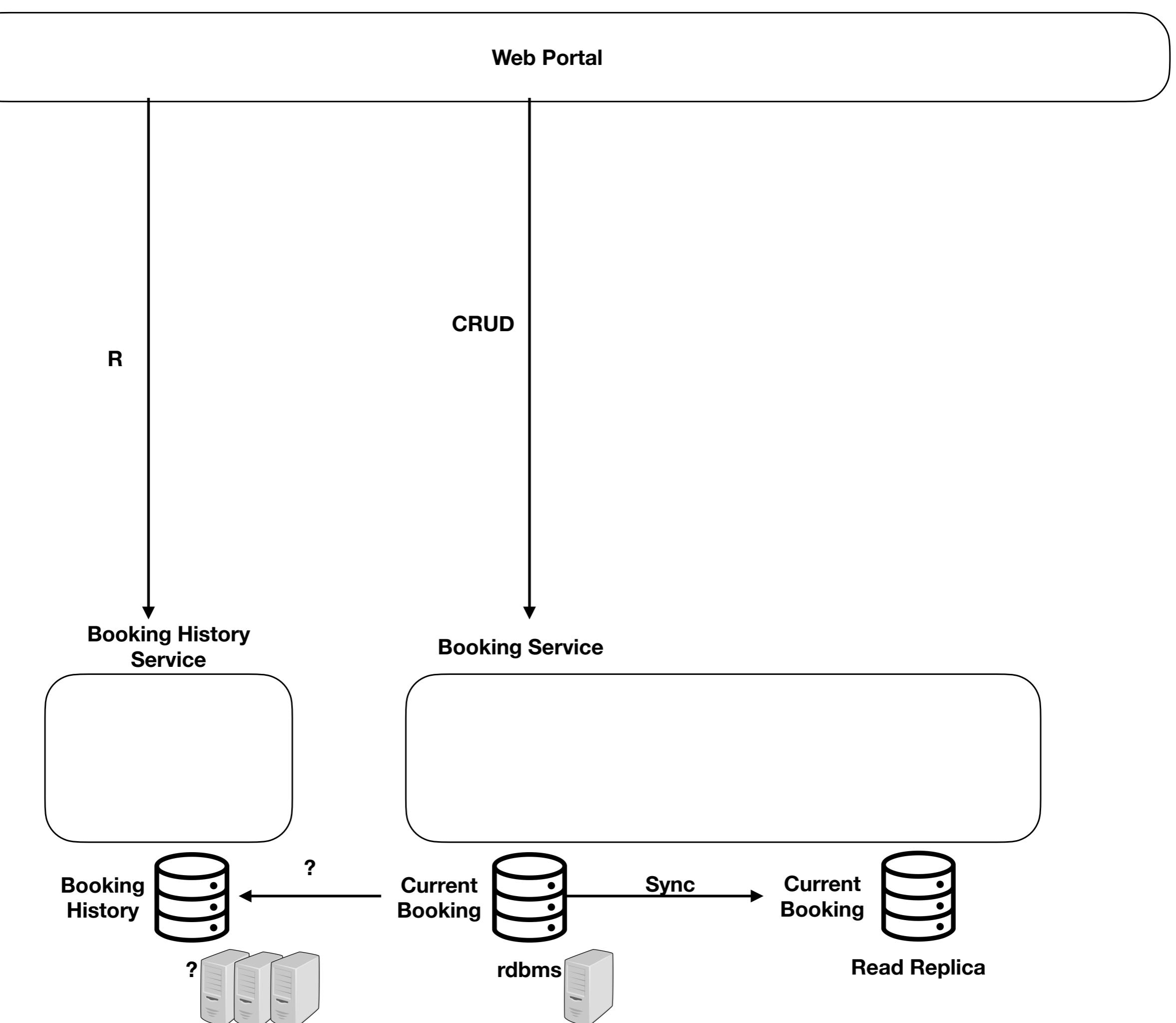
Listing Search Service

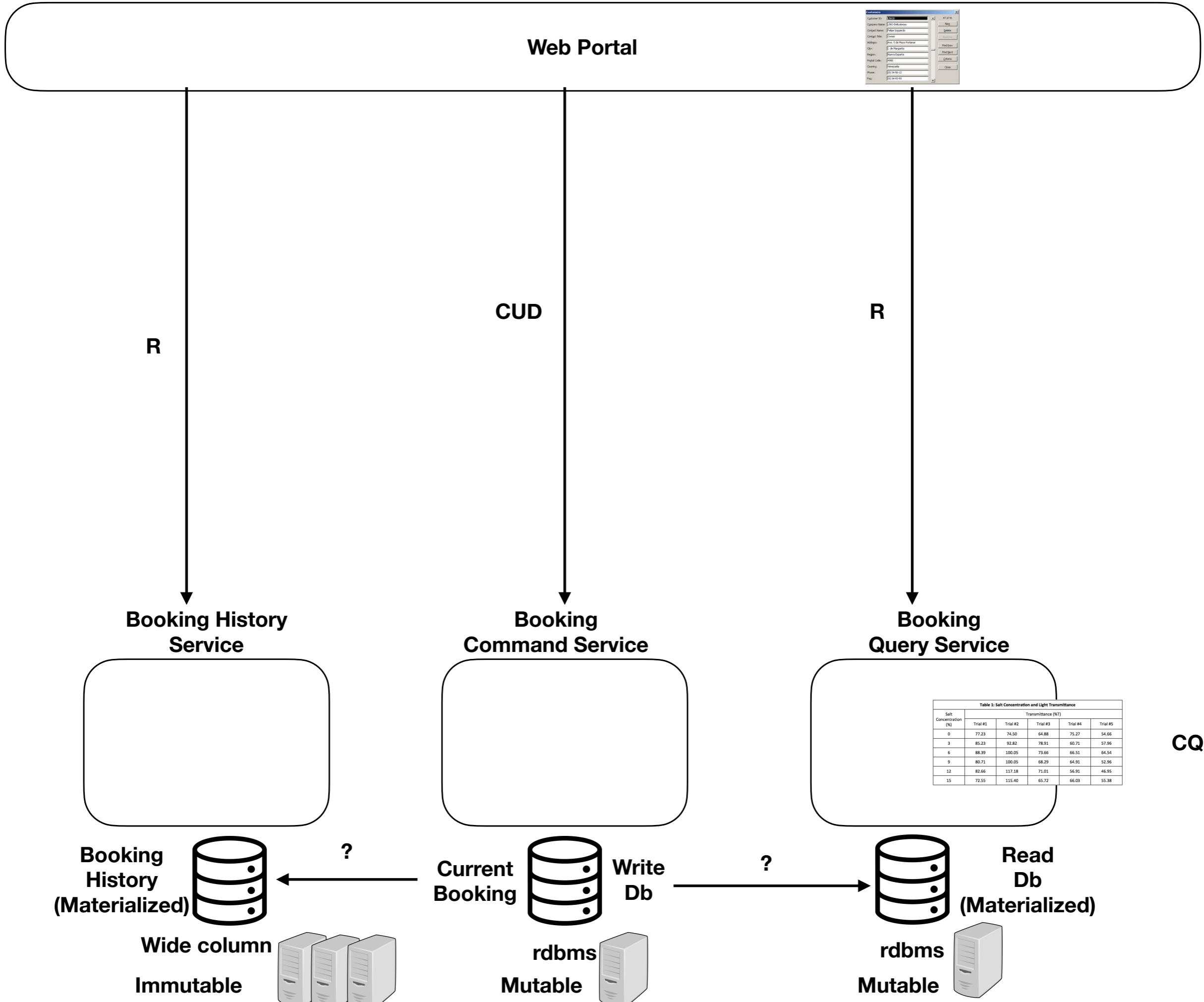


Text db

Microservice ?
EDA ?







TABLE_BOOK

Book ID	Genre ID	Price
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99

TABLE_GENRE

Genre ID	Genre Type
1	Gardening
2	Sports
3	Travel

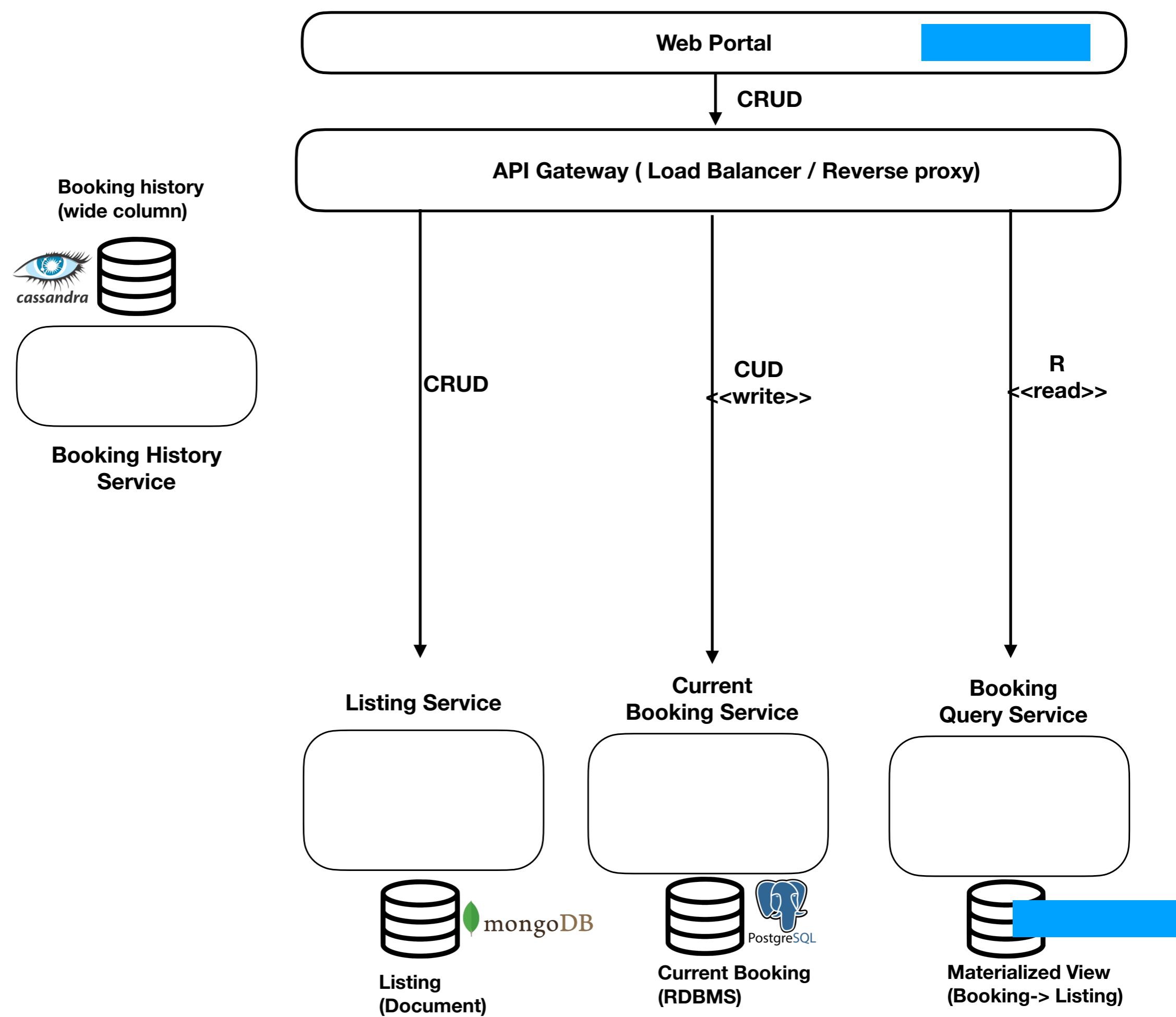
ID	NAME	SUBJECT	STATE	COUNTRY
29	Lalita	English	Gujrat	INDIA
33	Ramesh	Geography	Punjab	INDIA
49	Sarita	Mathematics	Maharashtra	INDIA
78	Zayed	History	Bihar	INDIA

Write
Db**Vs**Read
Db**Normalization**

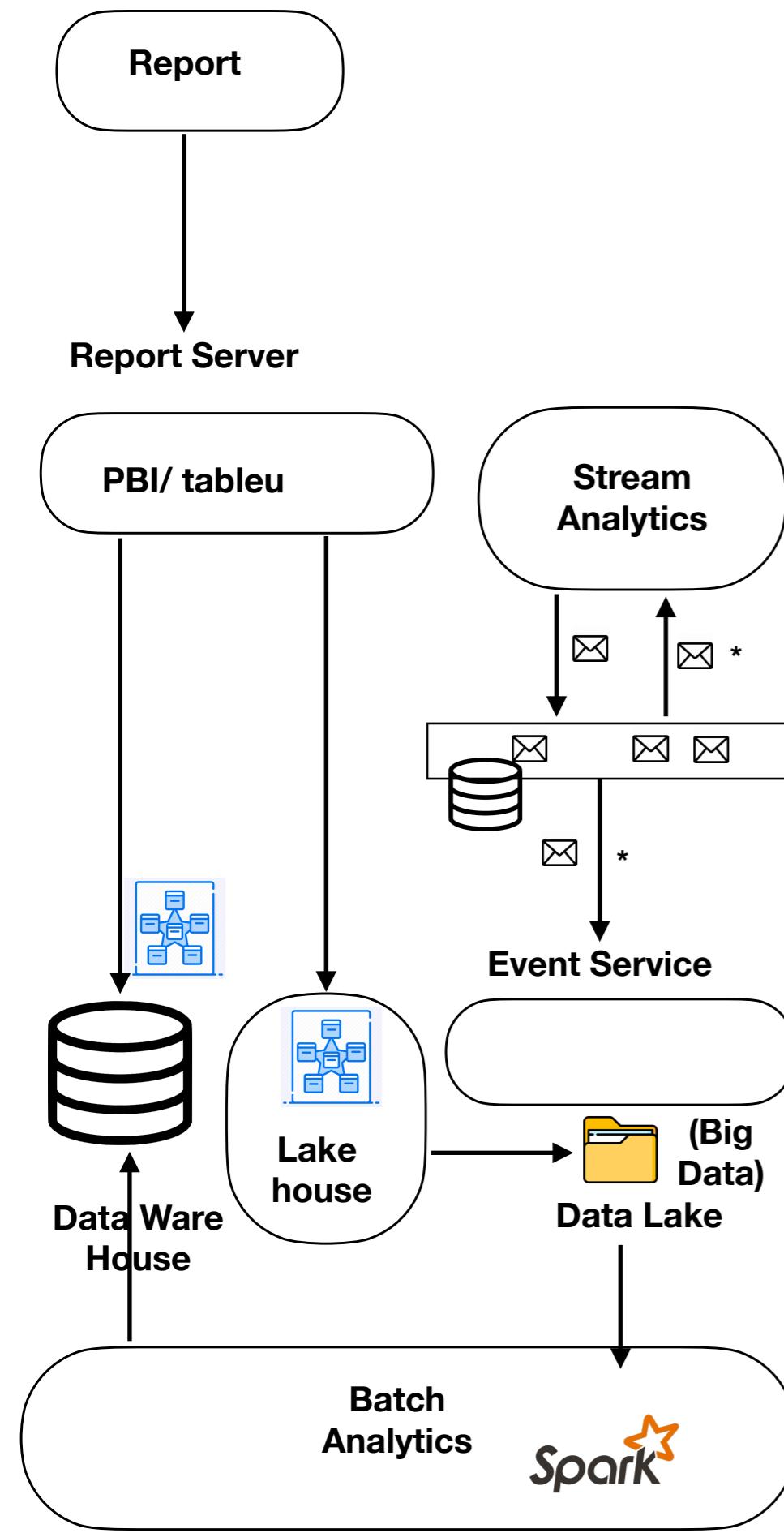
- **3rd normal form**
- **No duplicate**
- **Referential integrity**
- **Write friendly**

Denormalized

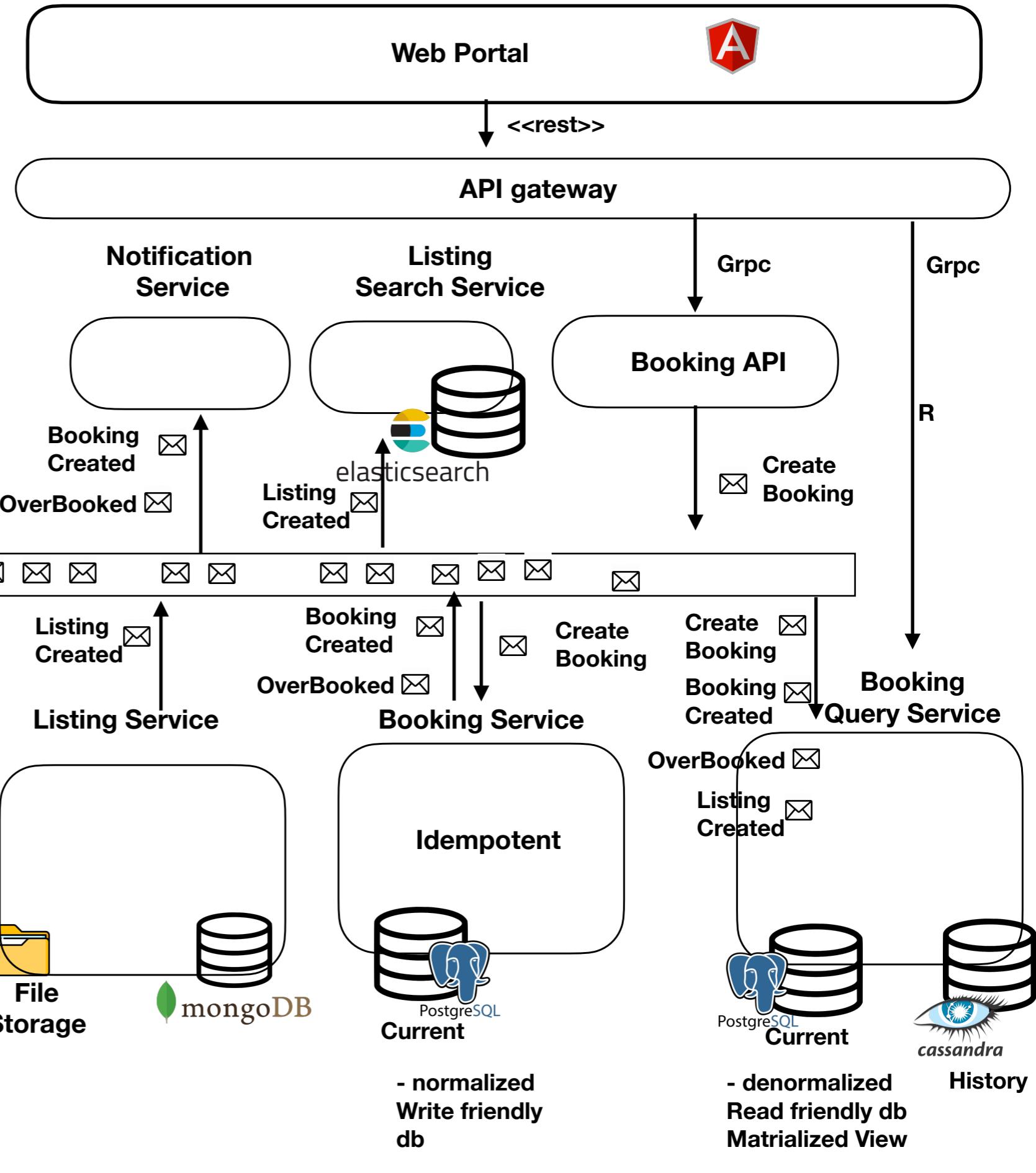
- **duplicate**
- **No joins**
- **Read friendly**



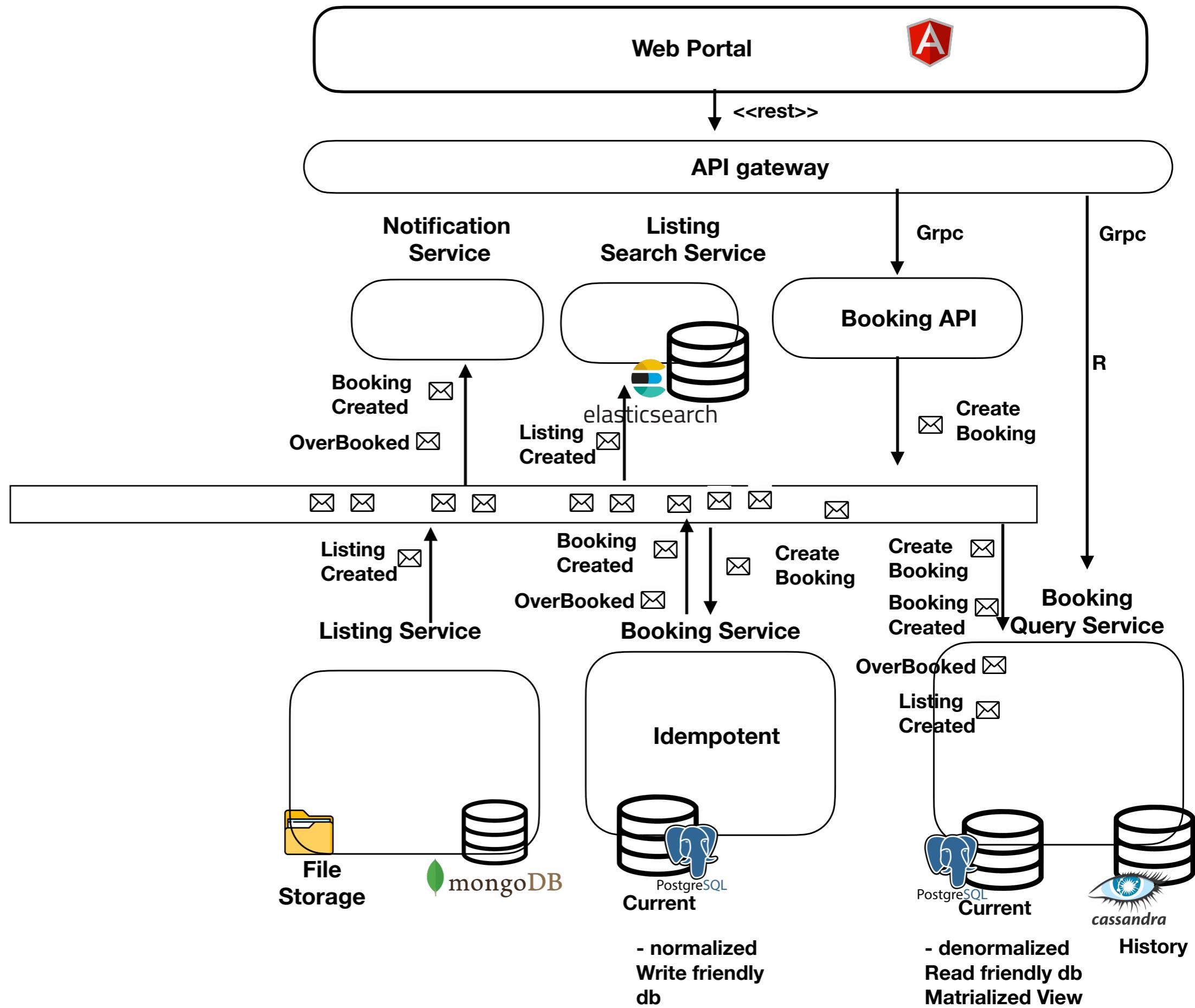
Manage Business



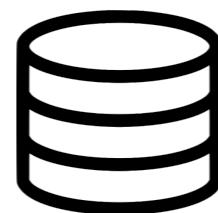
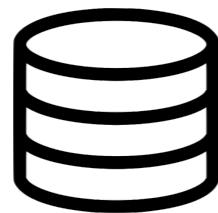
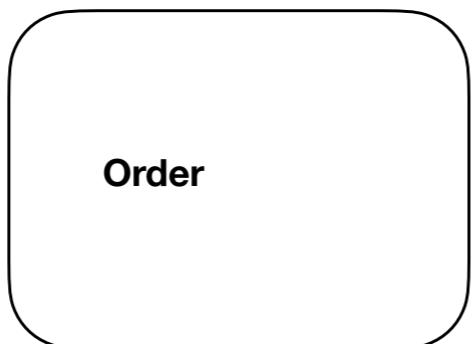
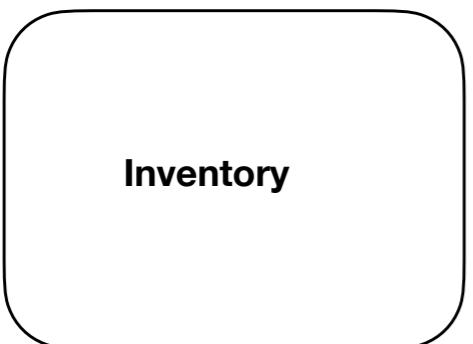
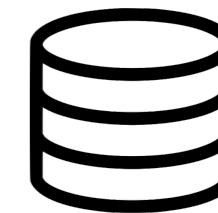
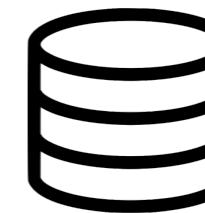
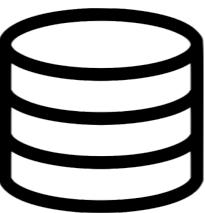
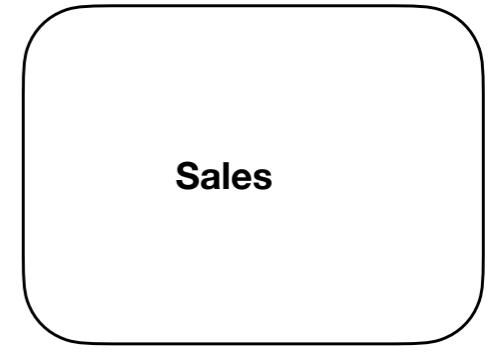
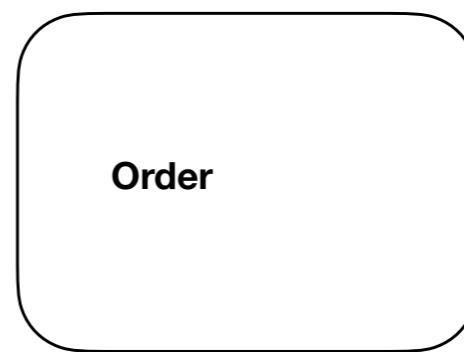
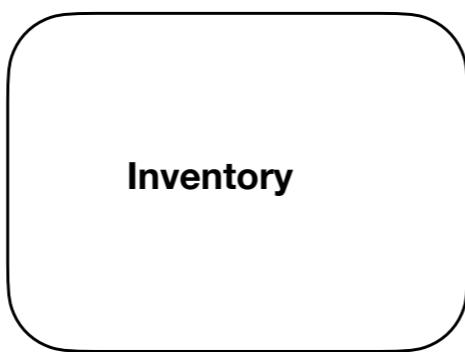
Run Business



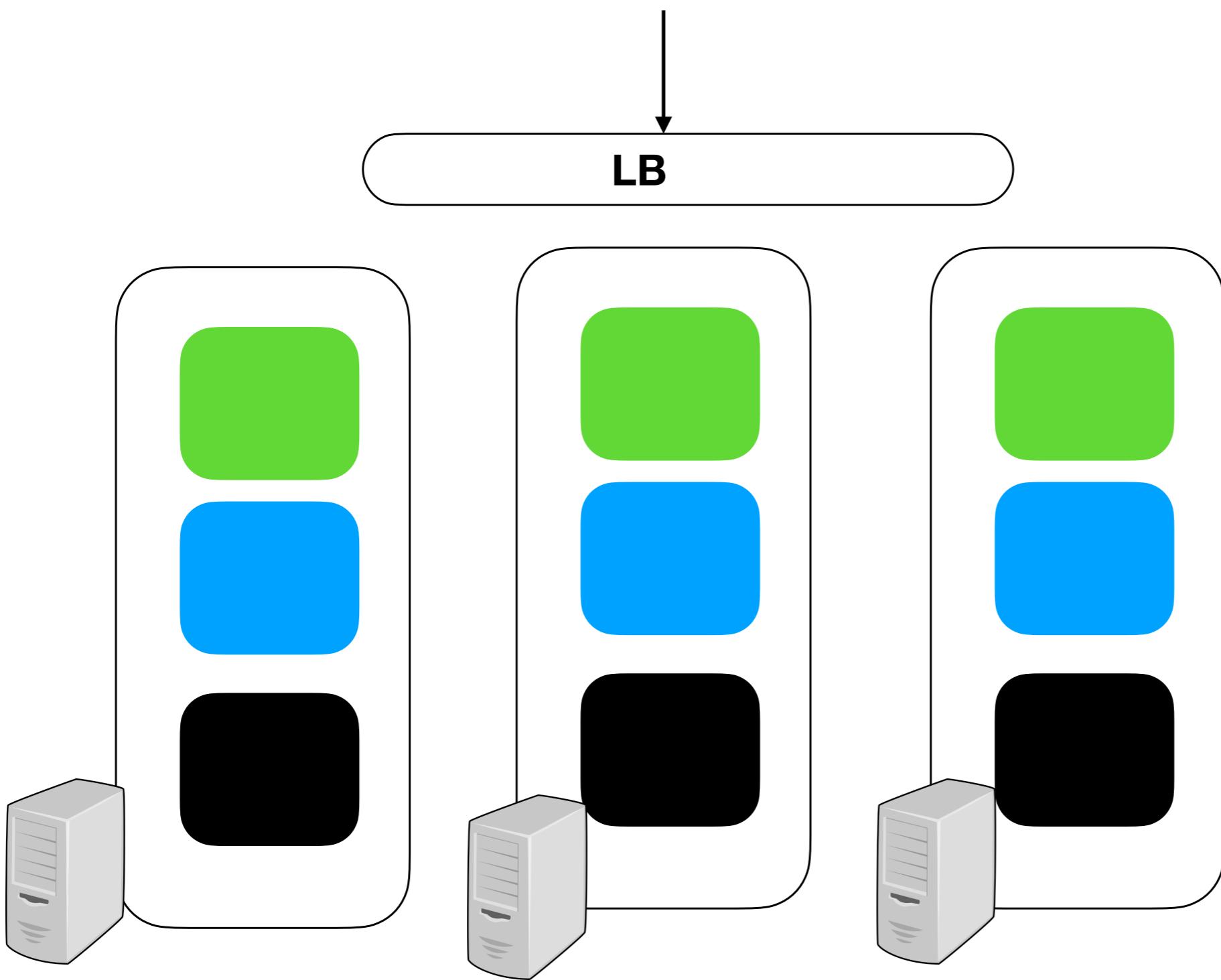
Run Business

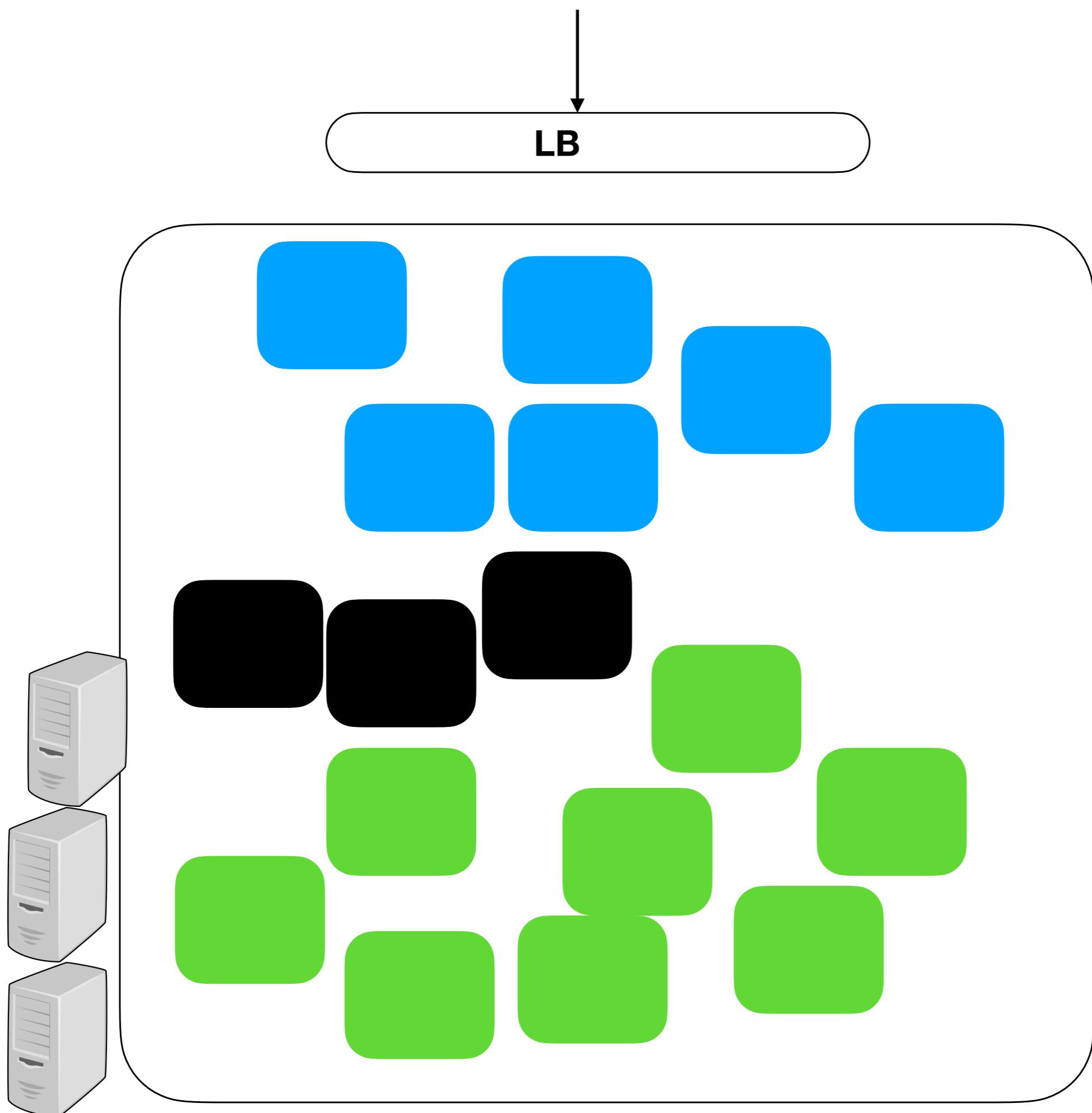


Consistency
Joins
Transactions

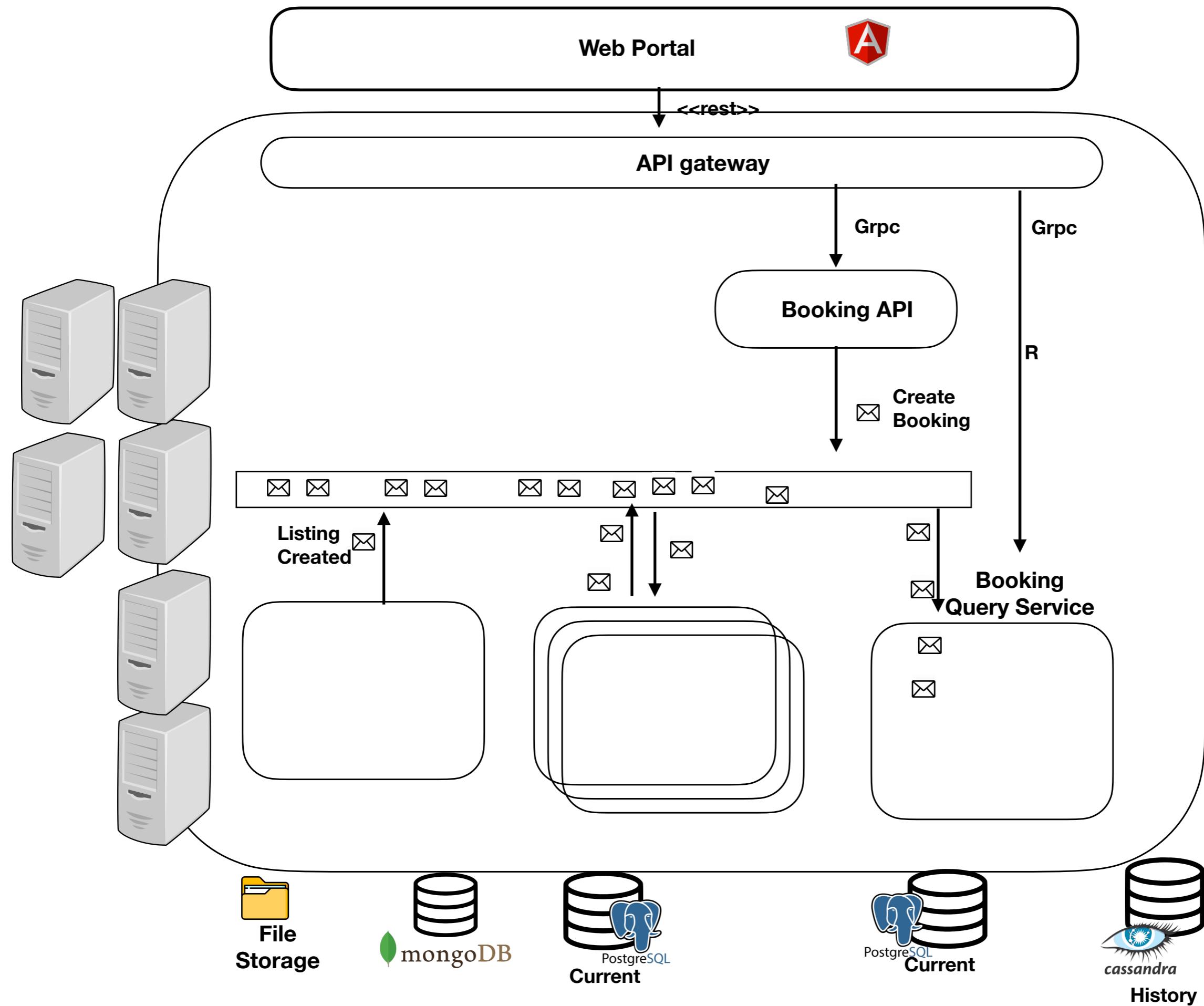


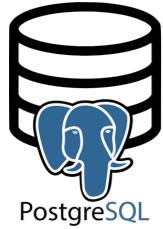
**Vertical slice/
Horizontal slice** **Vertical slice/
Horizontal slice**





Run Business





Read or write ?

De normalized Form

TABLE_BOOK_DETAIL

Book ID	Genre ID	Genre Type	Price
1	1	Gardening	25.99
2	2	Sports	14.99
3	1	Gardening	10.00
4	3	Travel	12.99
5	2	Sports	17.99

duplicates

no joins

read friendly

not write friendly

3rd Normal Form

TABLE_BOOK

Book ID	Genre ID	Price
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99

TABLE_GENRE

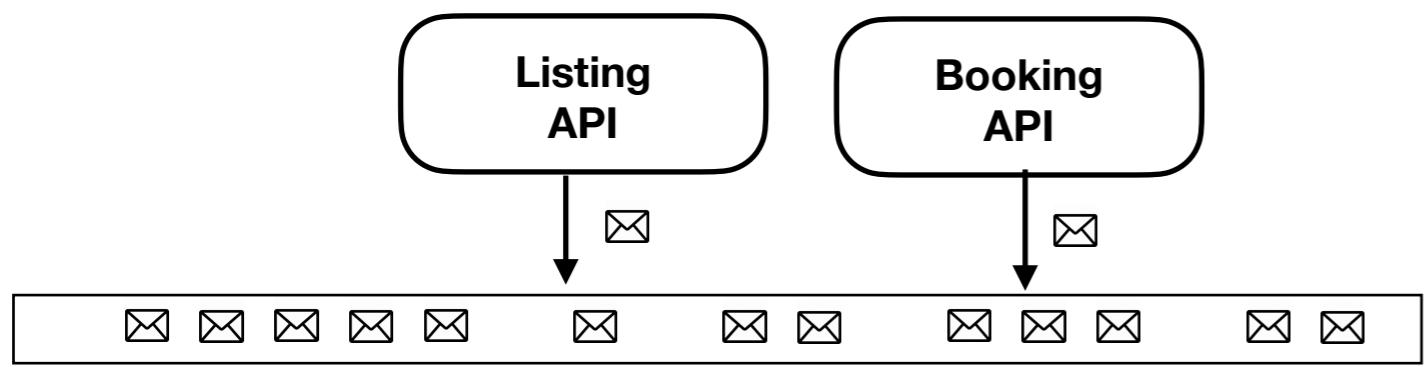
Genre ID	Genre Type
1	Gardening
2	Sports
3	Travel

no duplicates

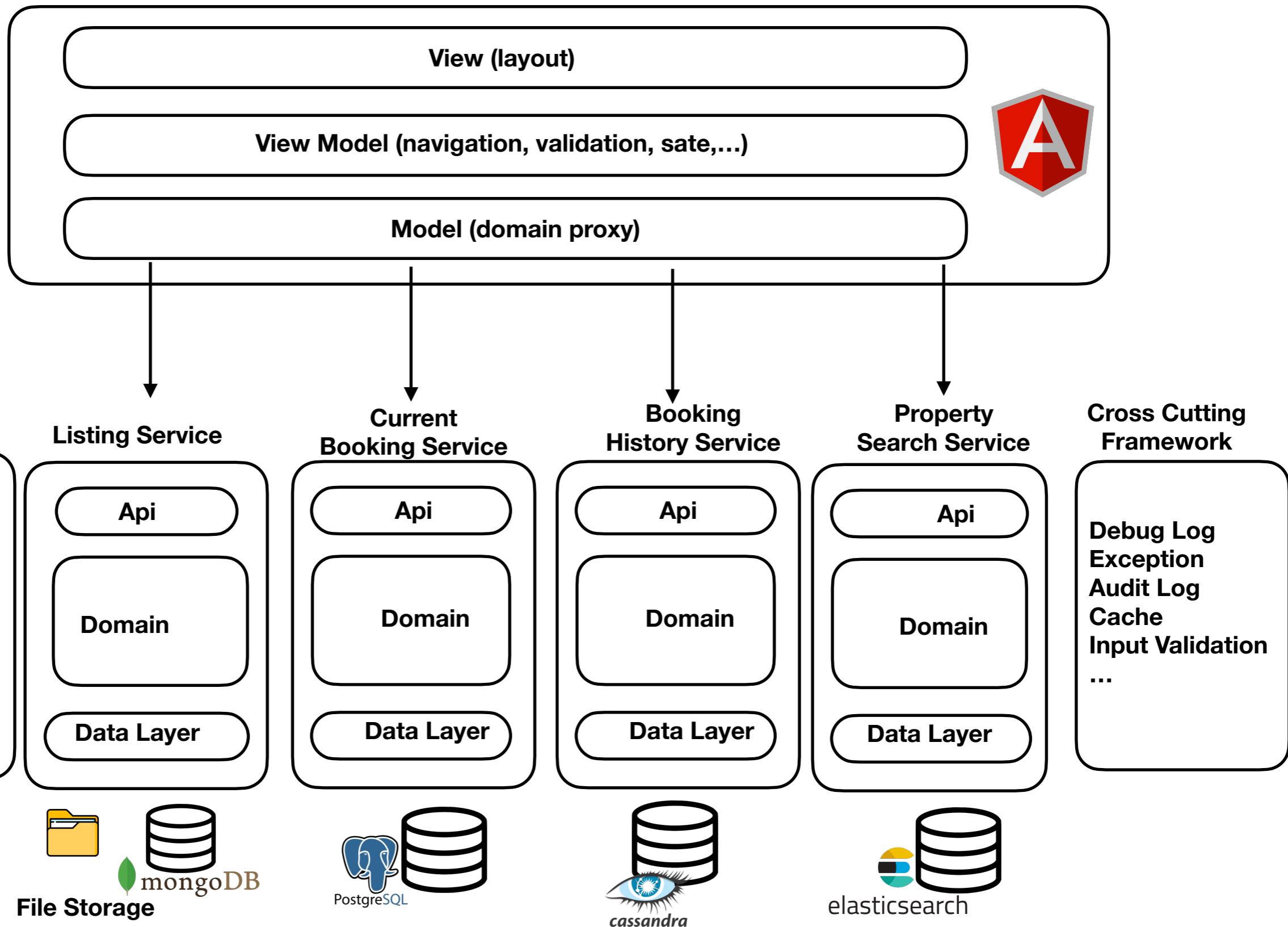
joins

write friendly

not read friendly



Web Portal



API

*** completes in next few seconds**

Immediate

Pessimistic

**# redbus
book my show**

Optimistic

irctc

*** locks before
transaction**

Messaging

*** completes in next few minutes/hours**

Eventual

*** locks within
transaction**

Data

Application Type

Java, Native, Web, Mainframe, Electron,

Functionality Type

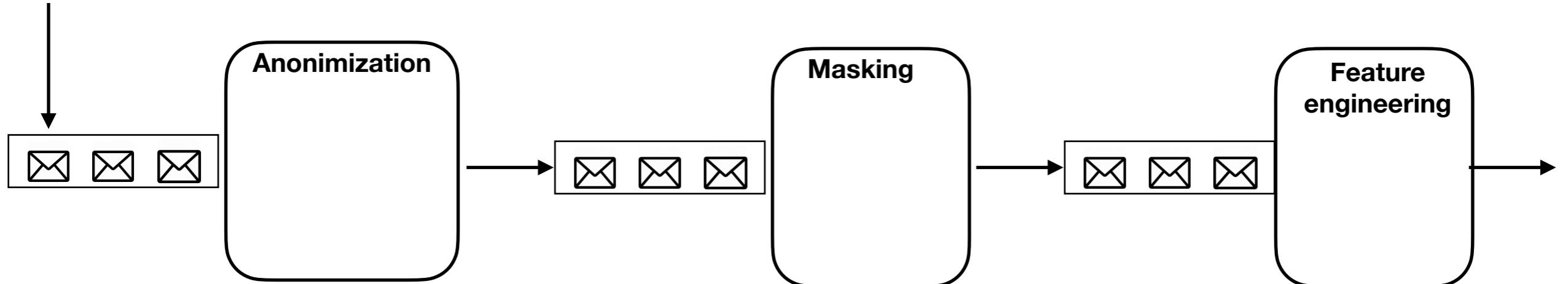
Anonimization, Masking, Feature engineering, ...

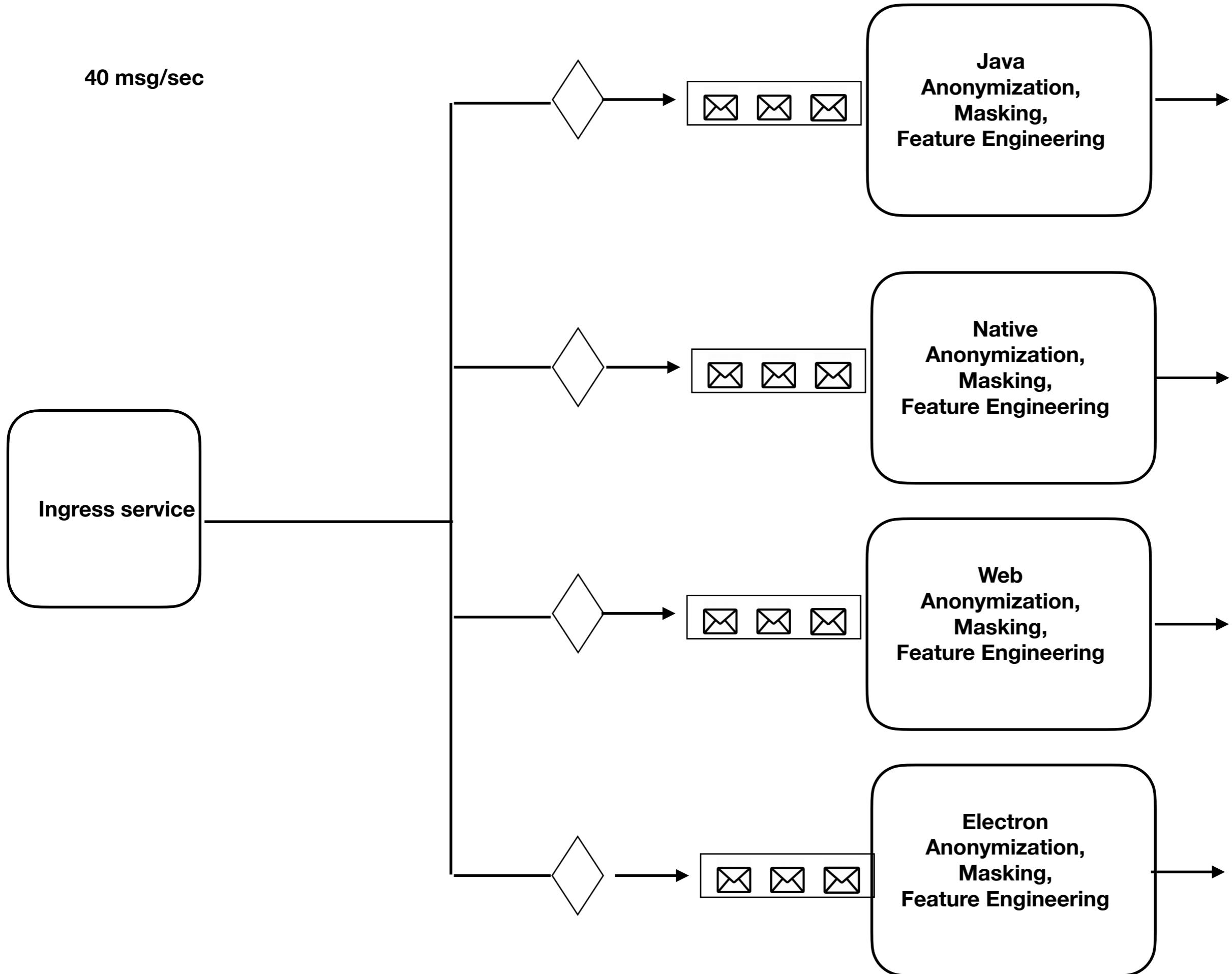
Java,
Anonymization

Java
Anonymization,
Masking,
Feature Engineering

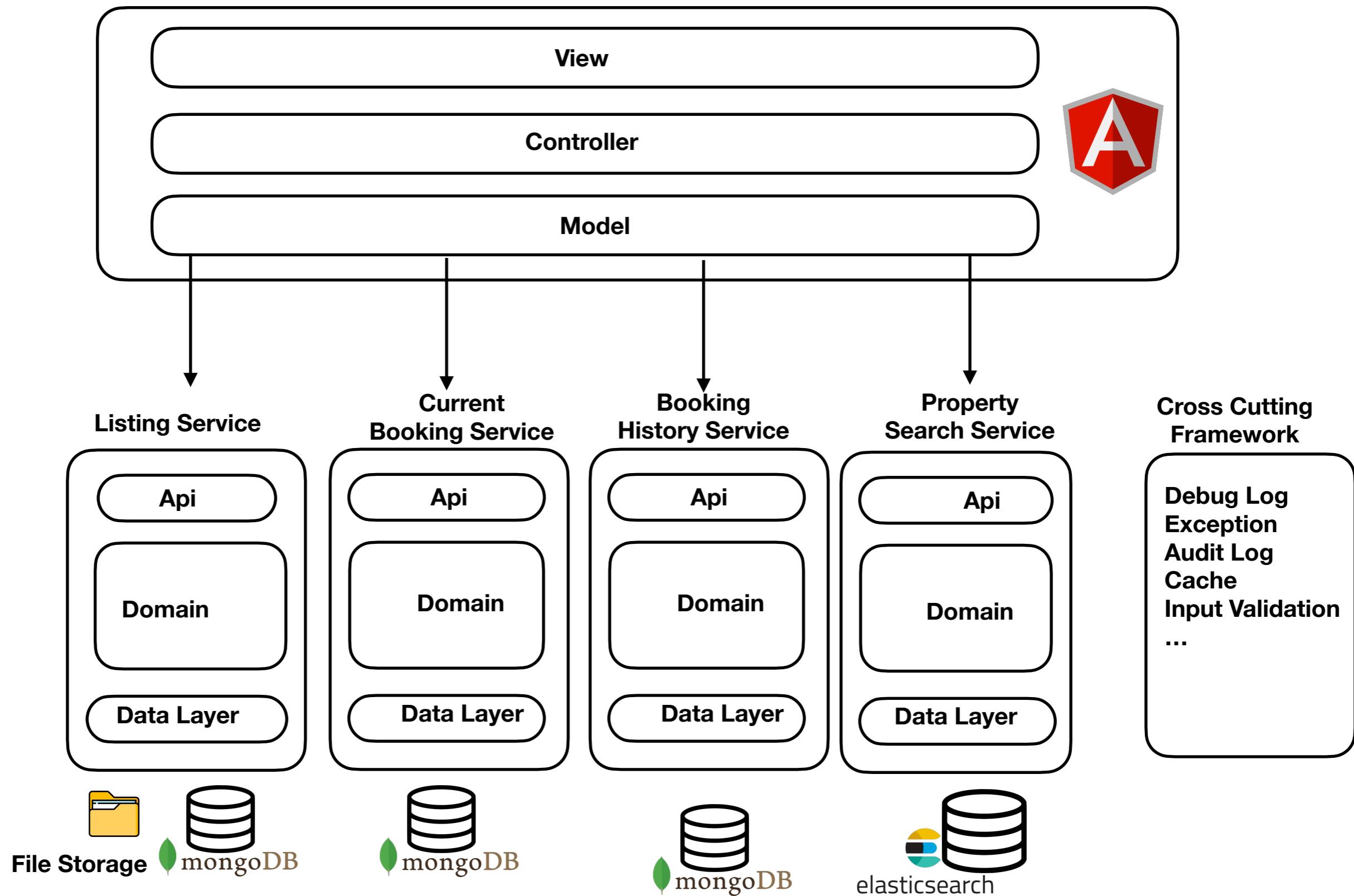
Anonymization
Java,
Native,
Web
Mainframe,
Electron

8 msg/sec

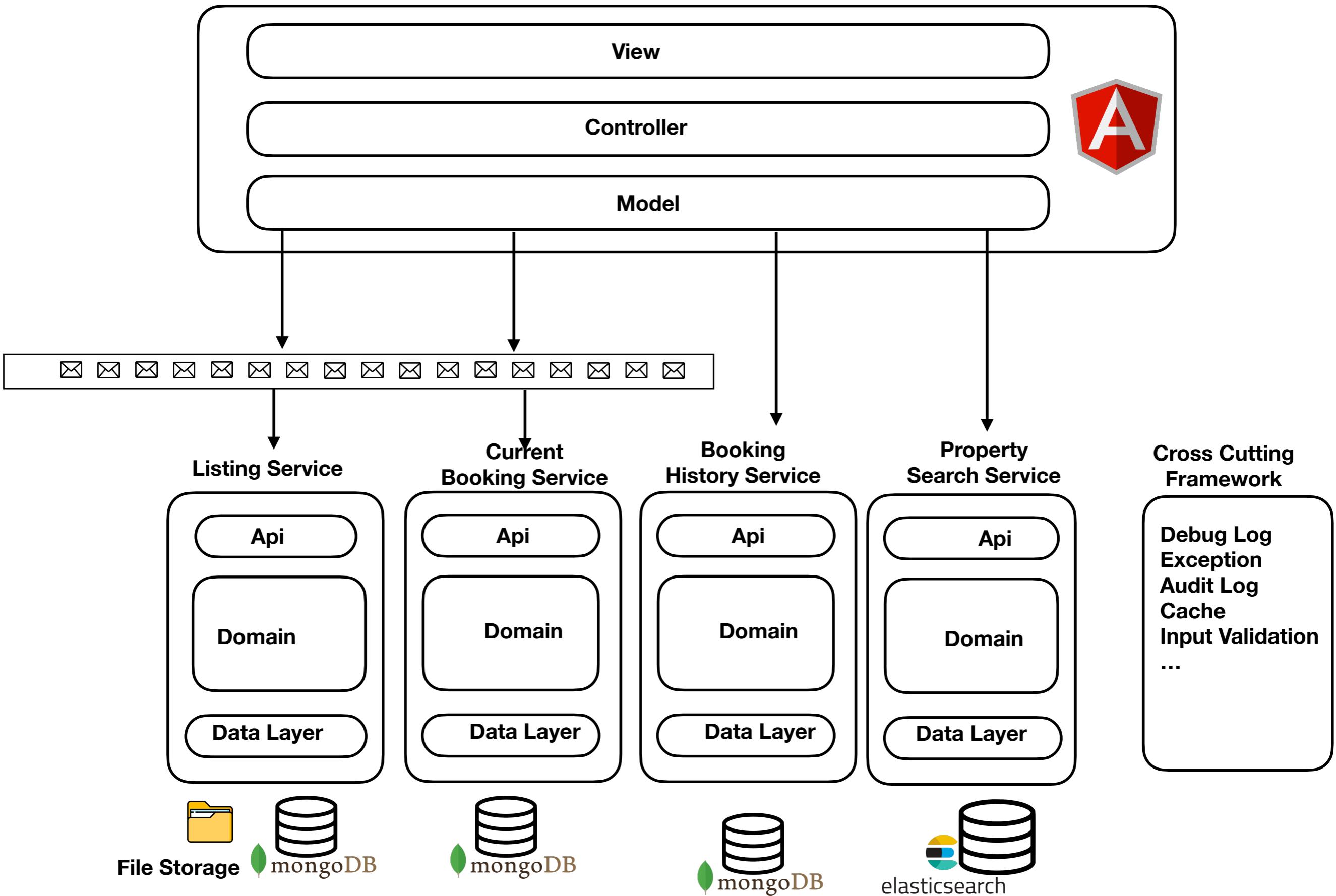


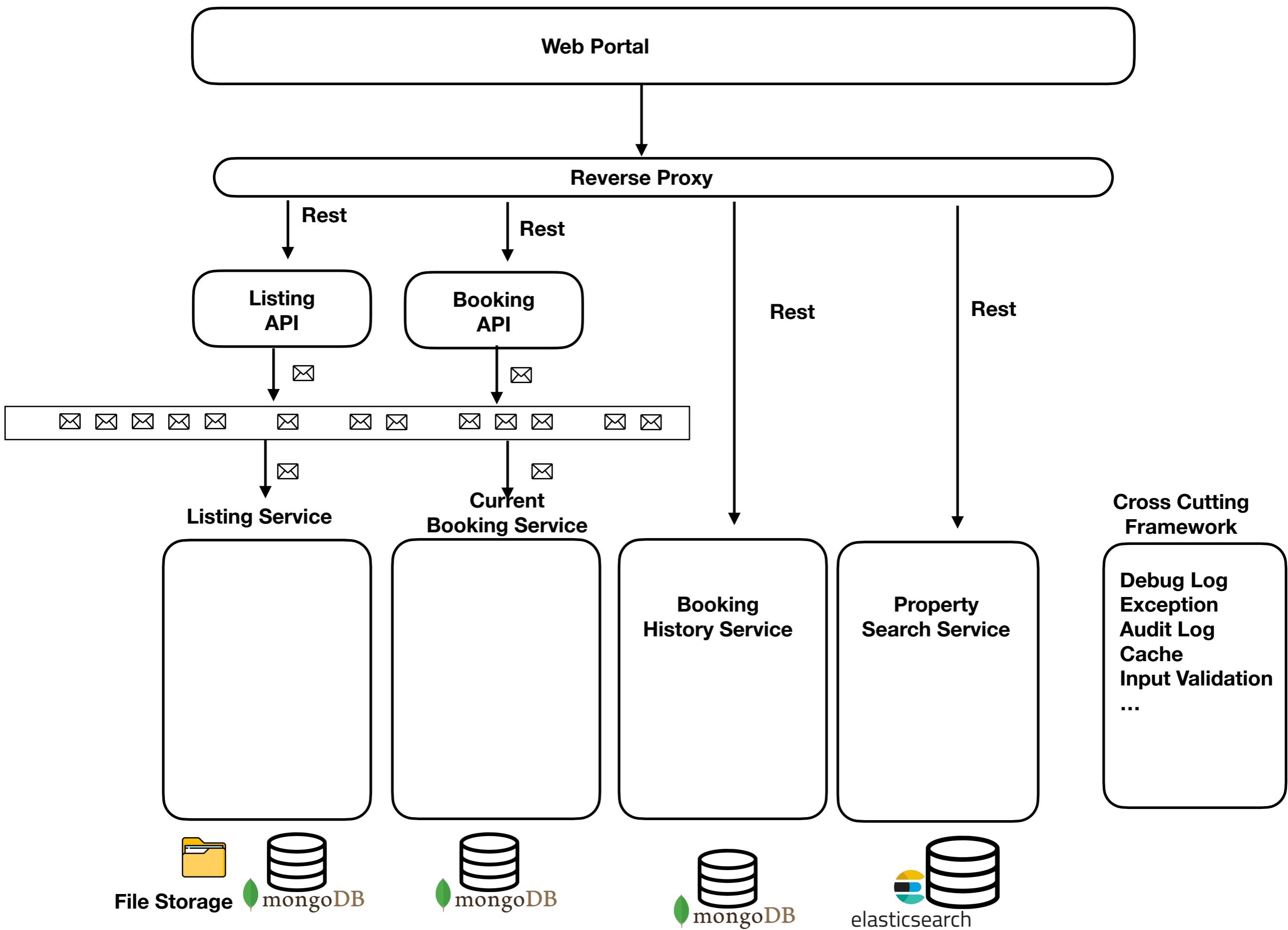


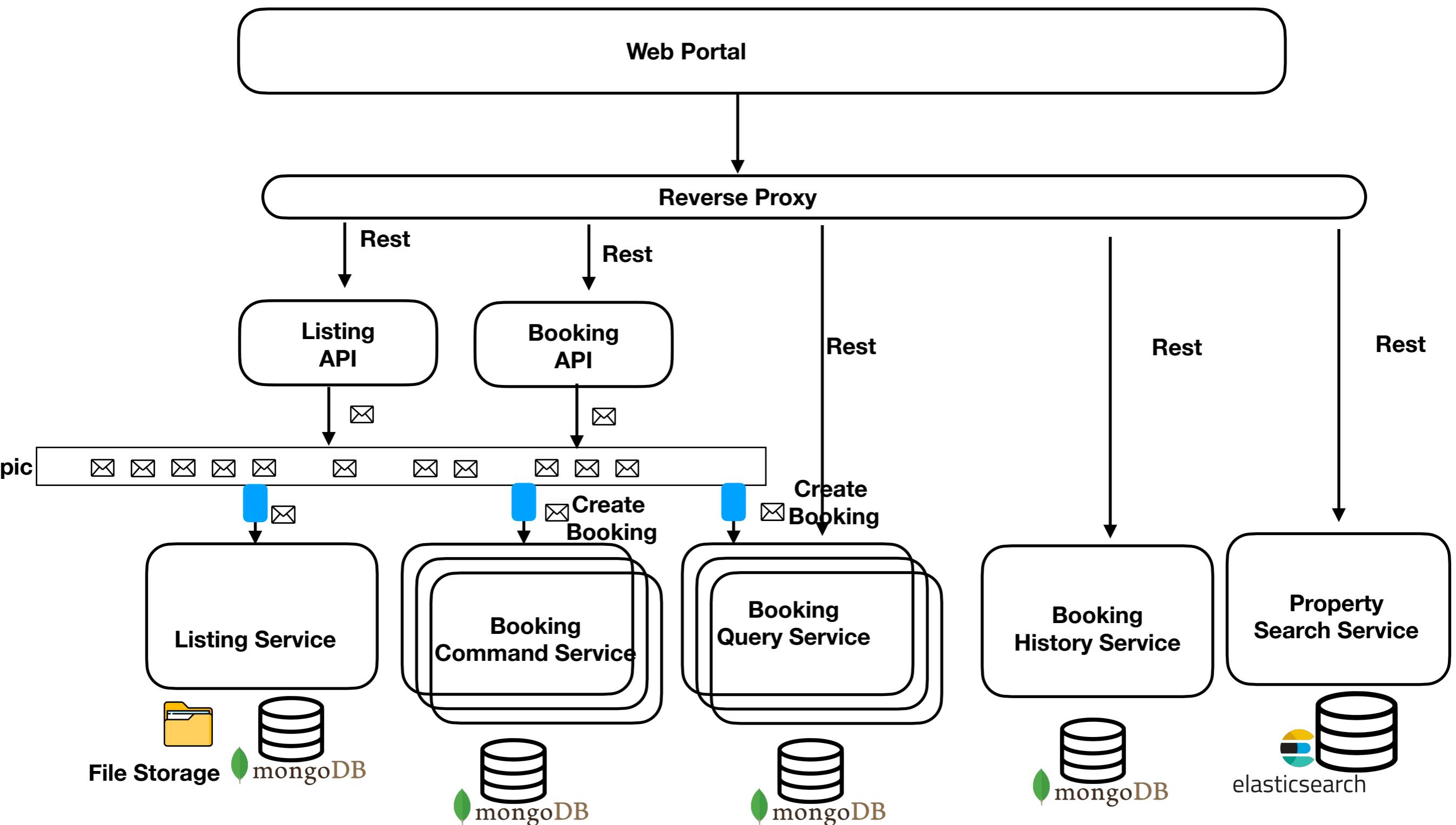
Web Portal



Web Portal

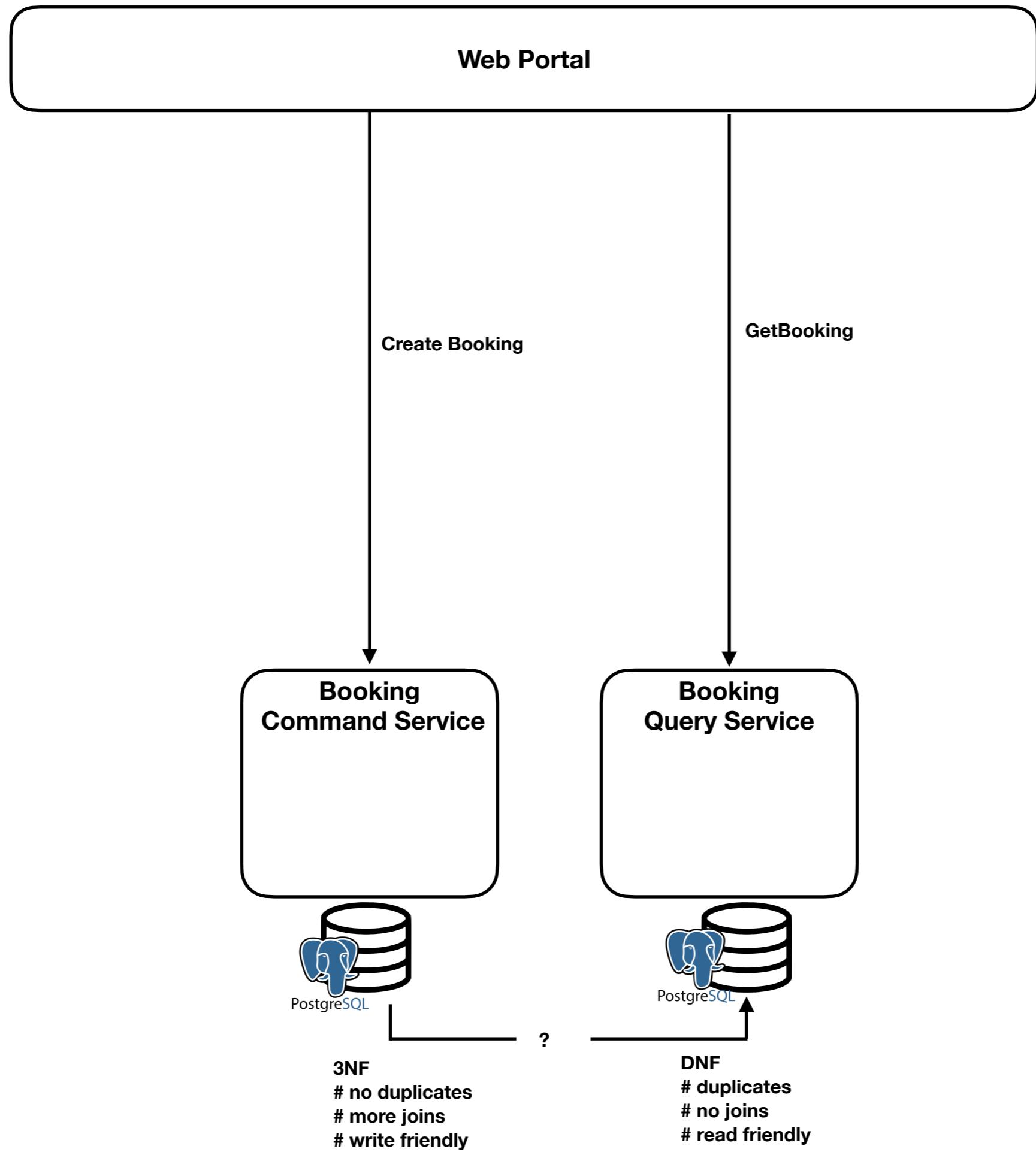


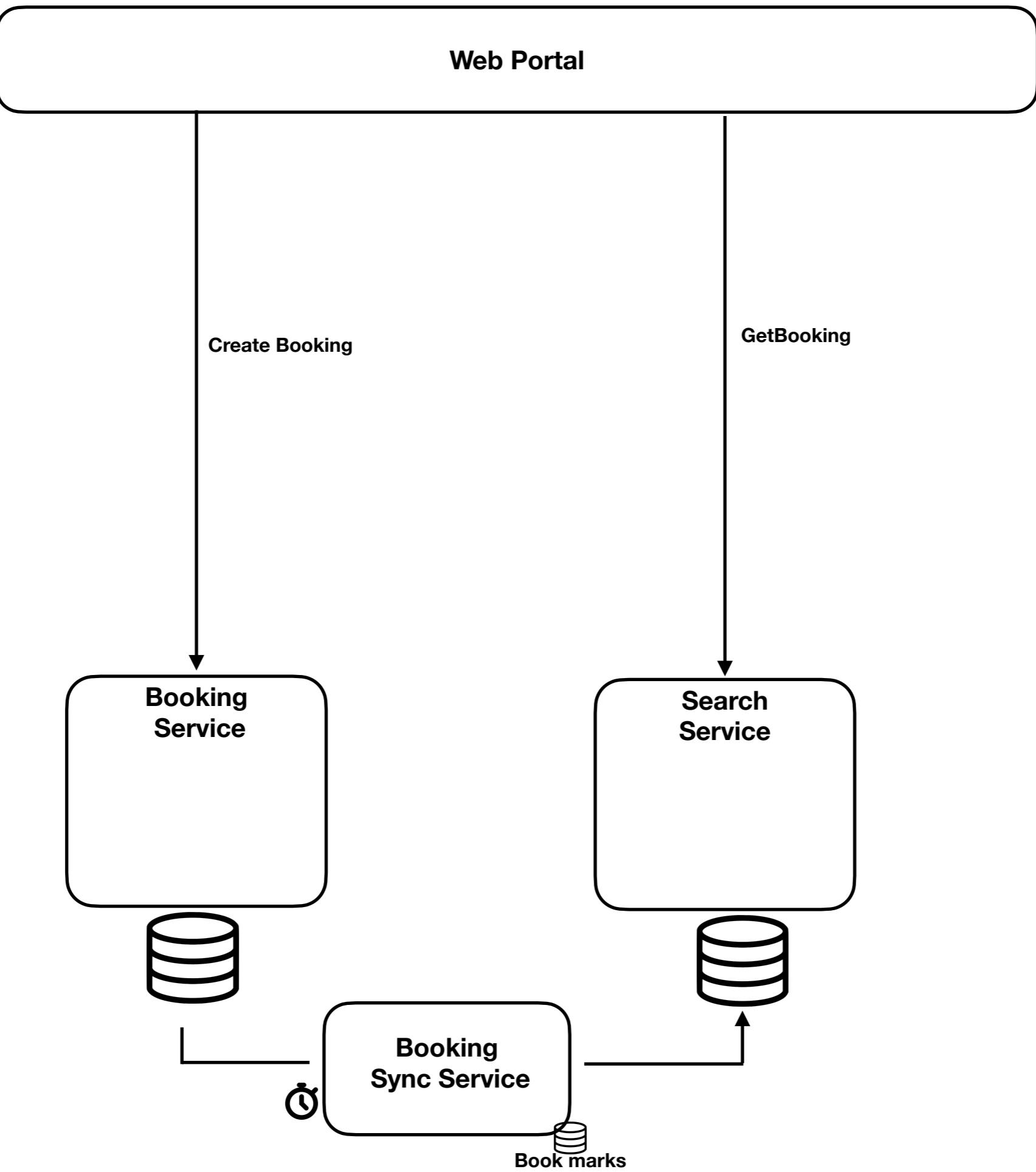


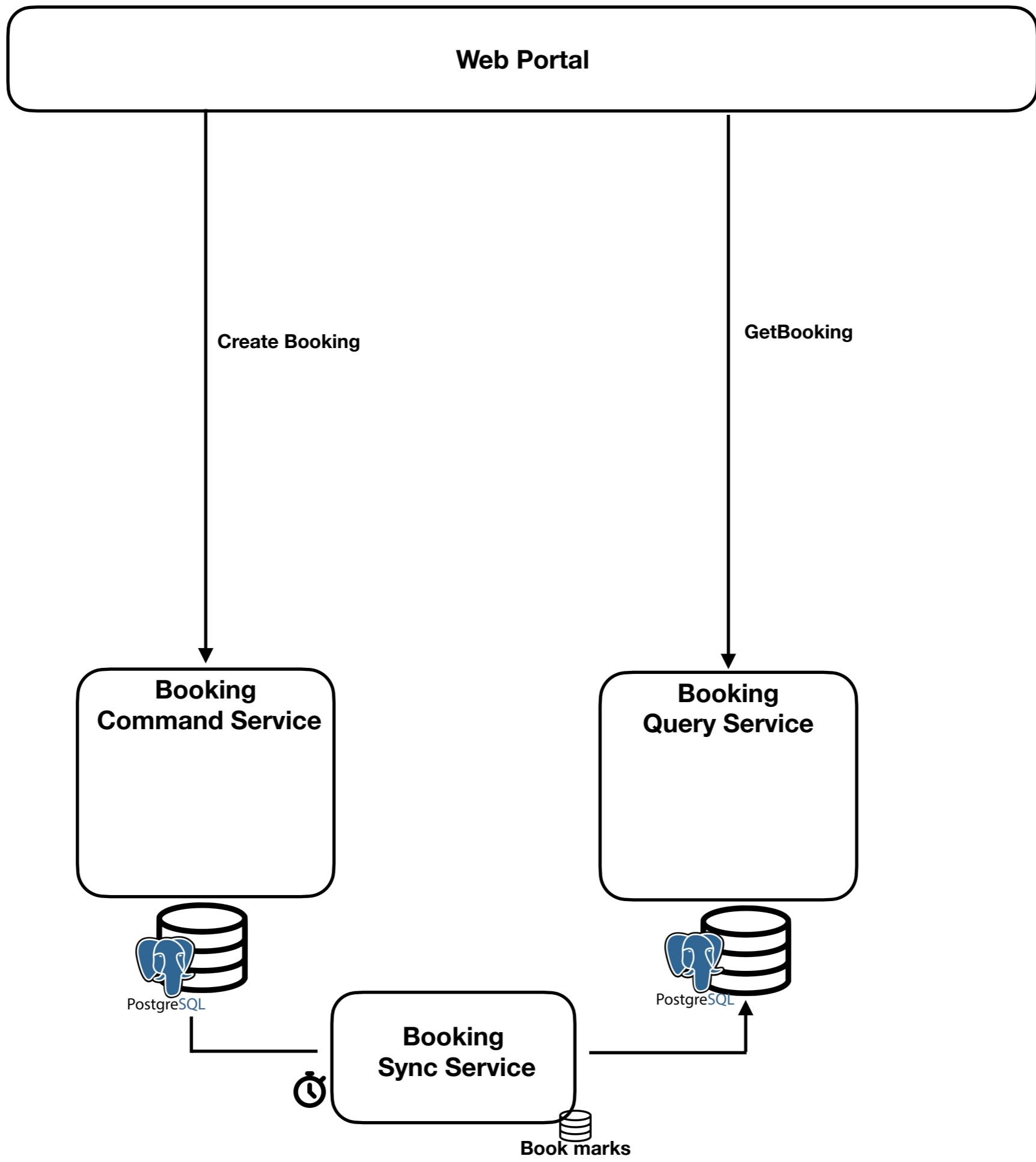


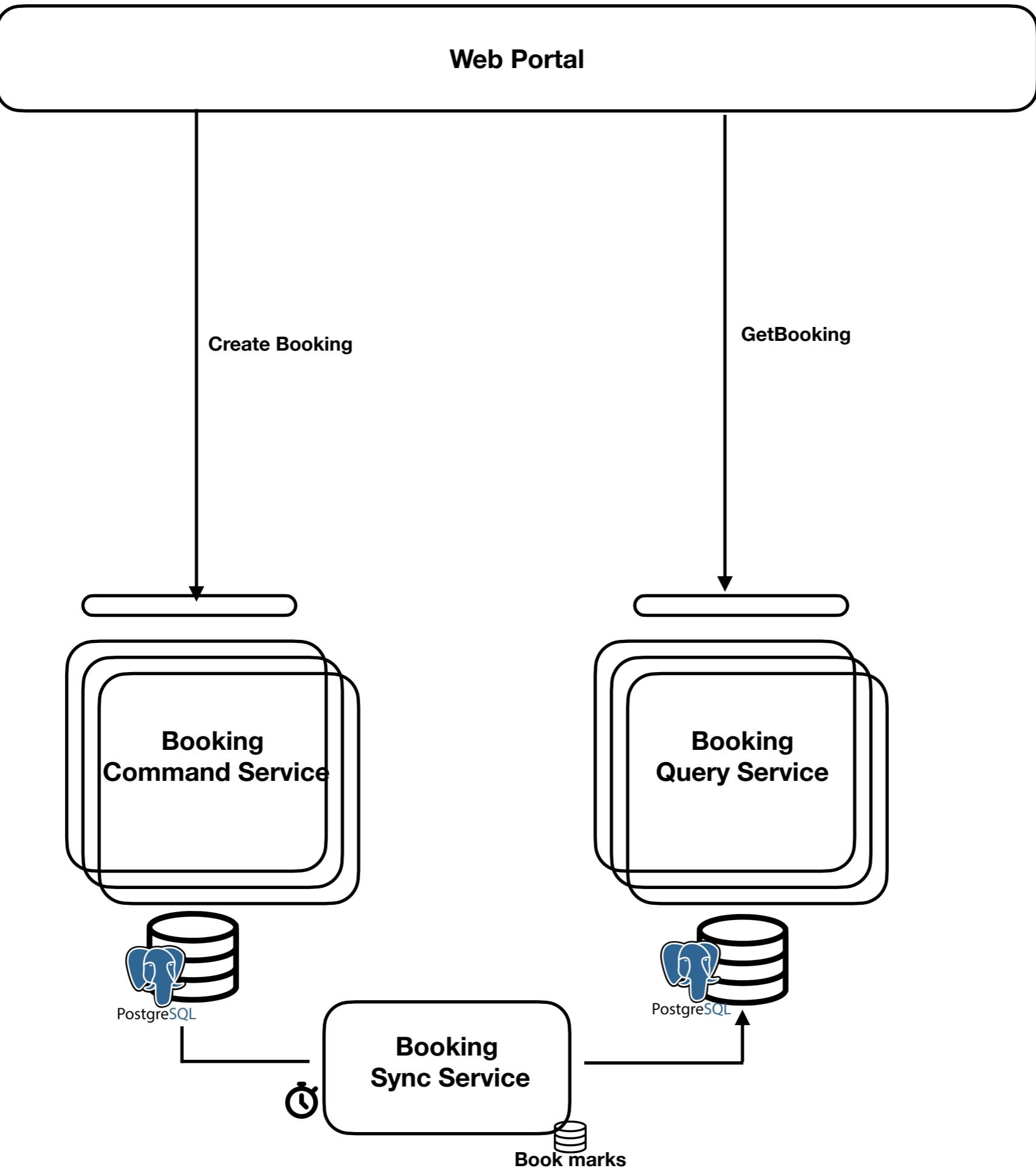
Service

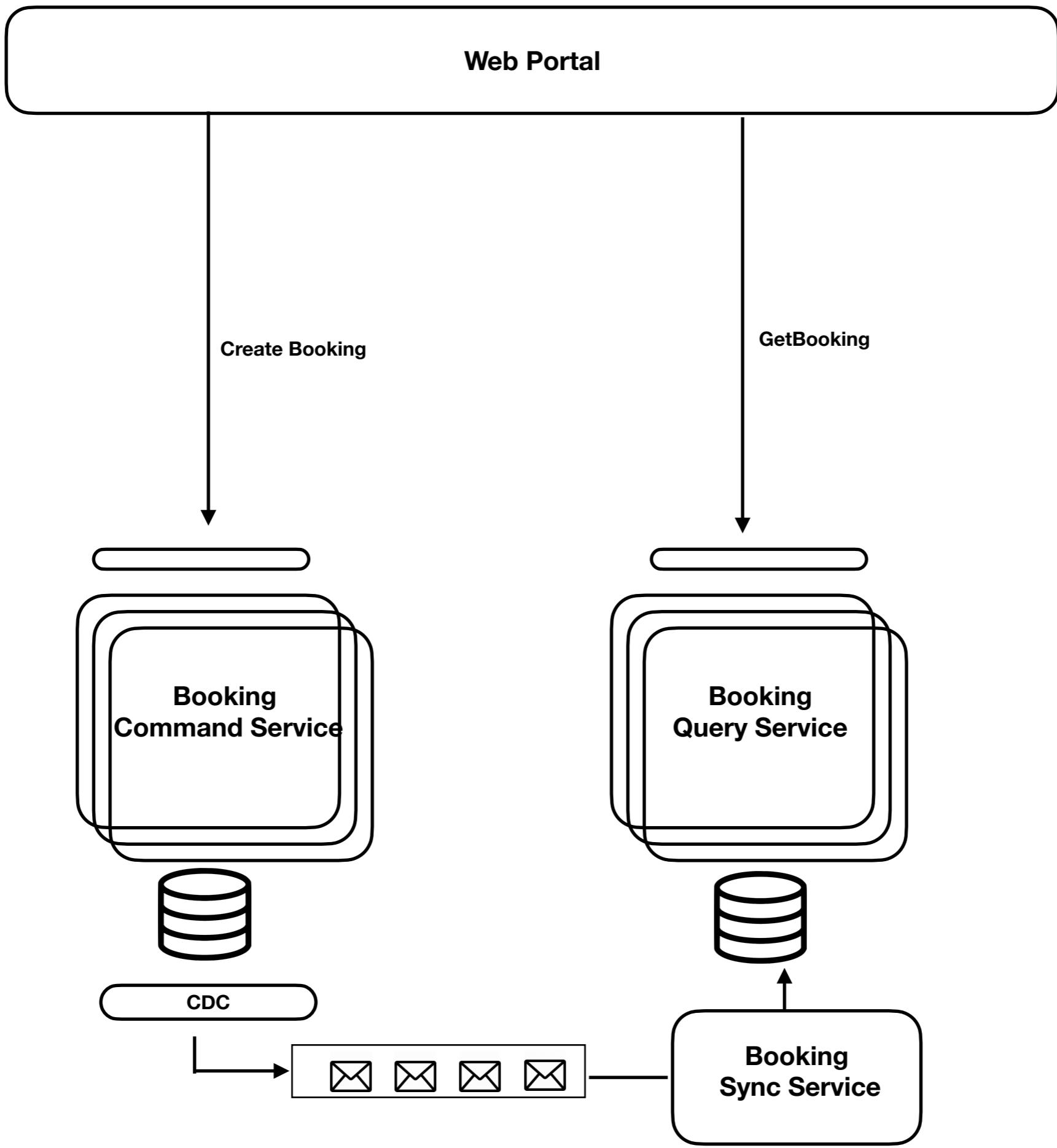
**# idempotency
unordered delivery
undoable operation**

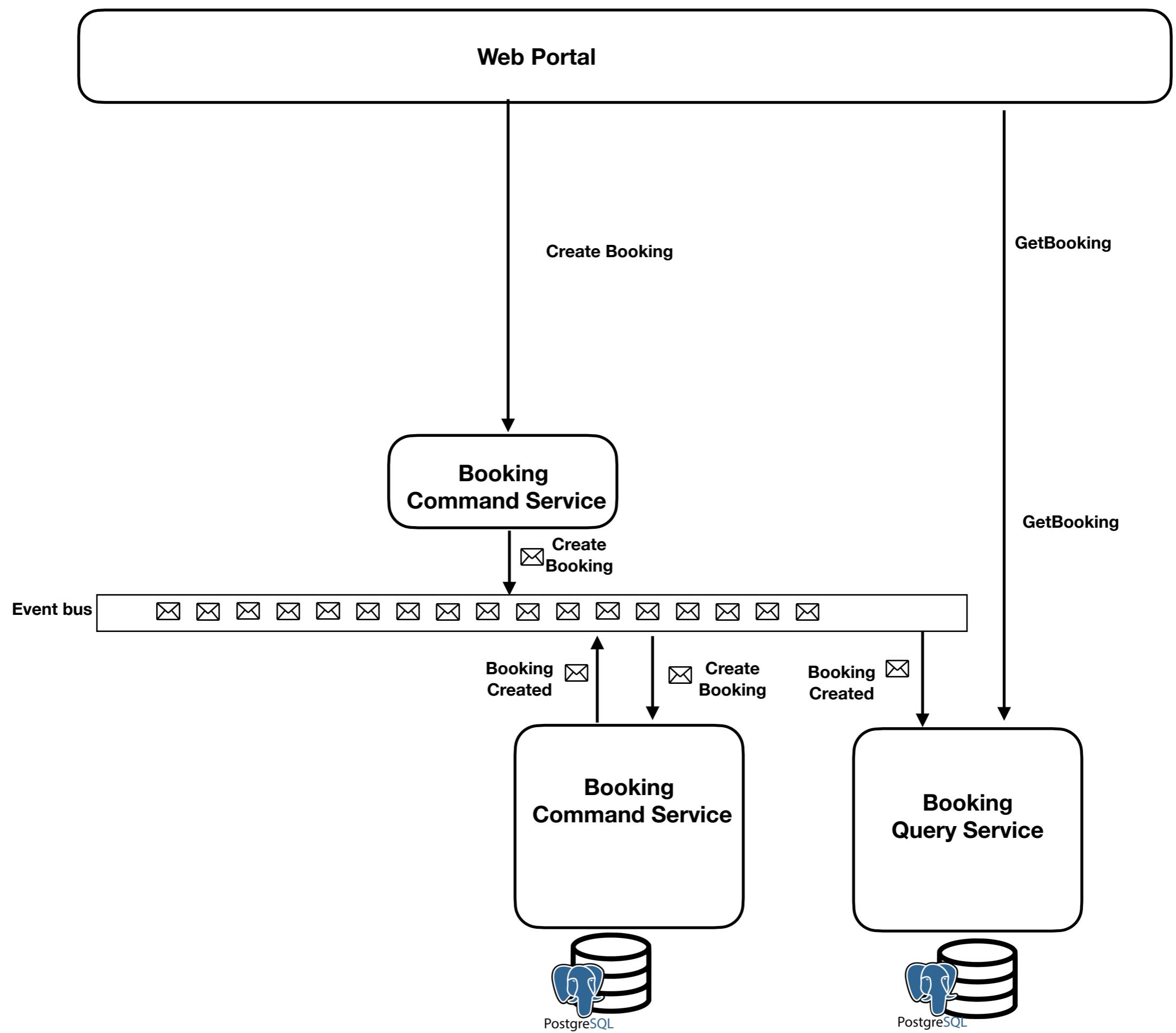


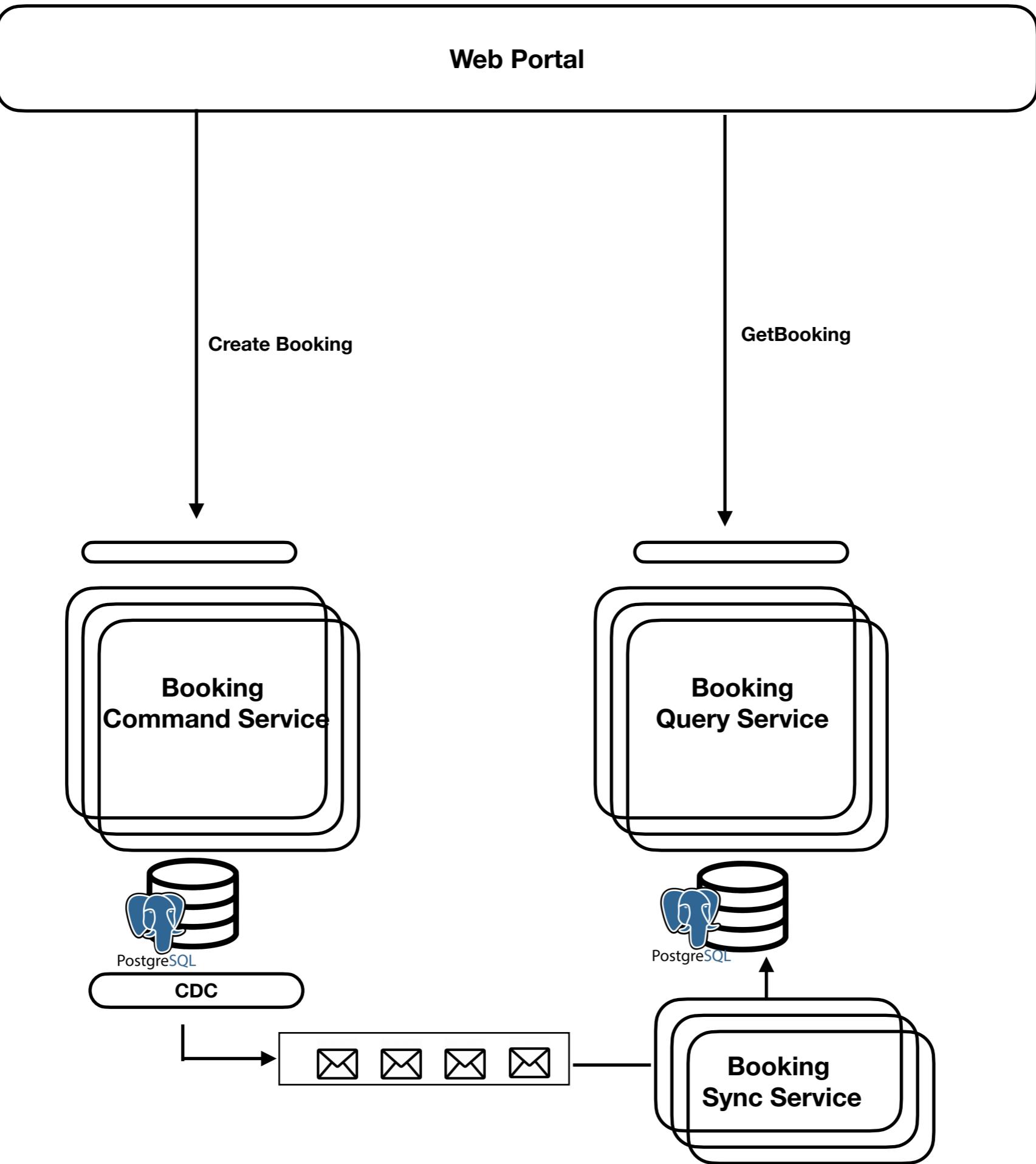


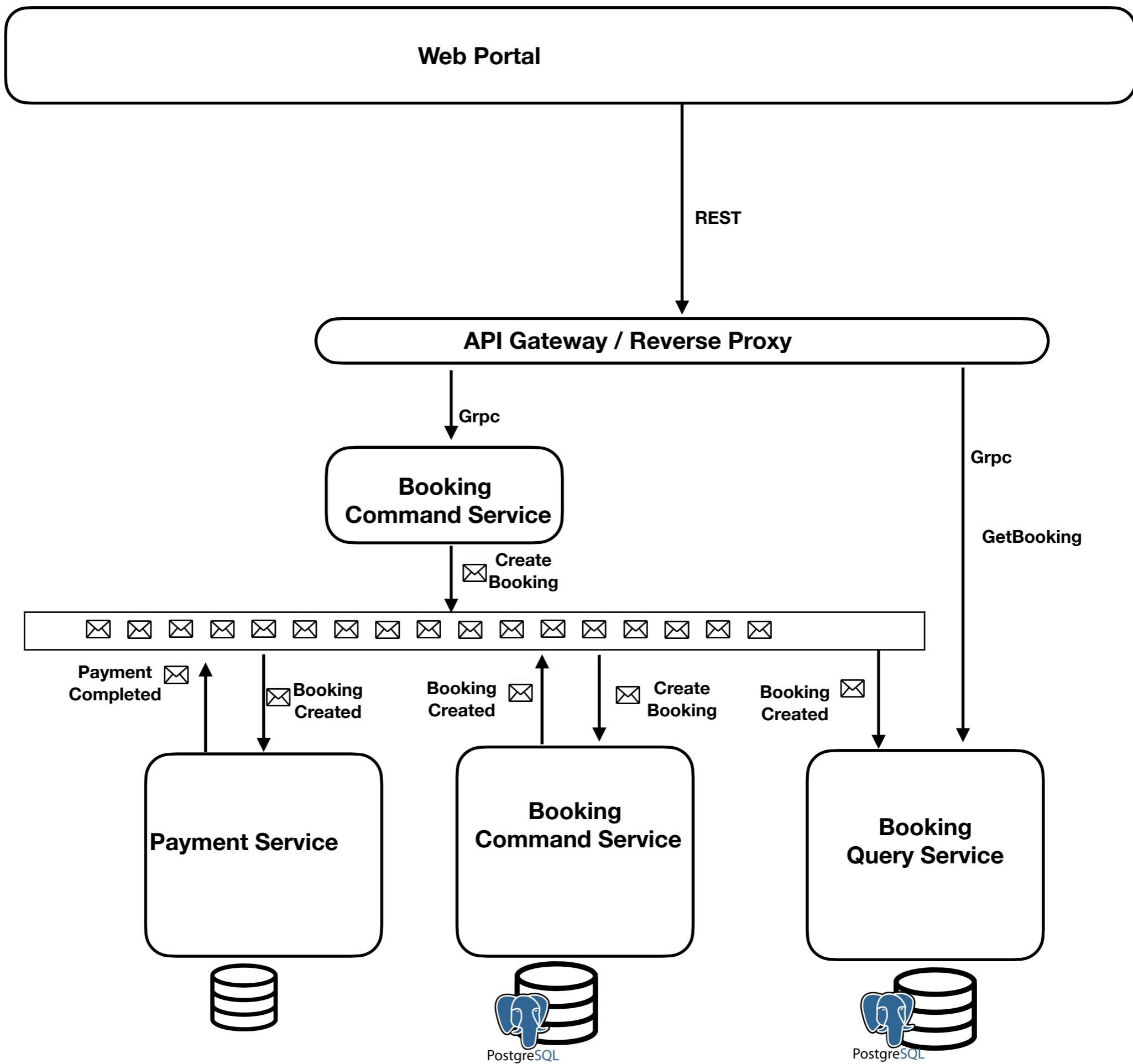




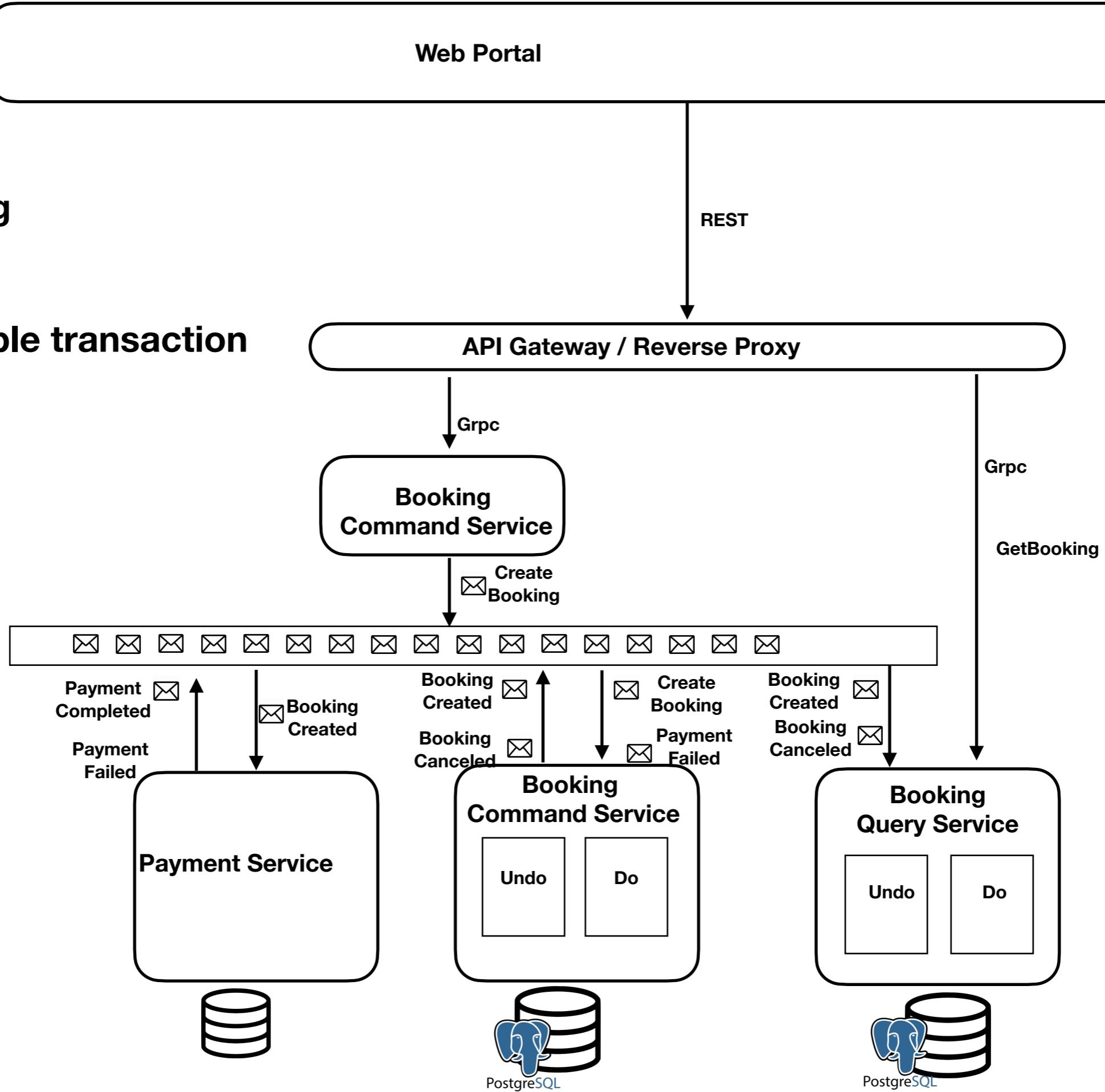


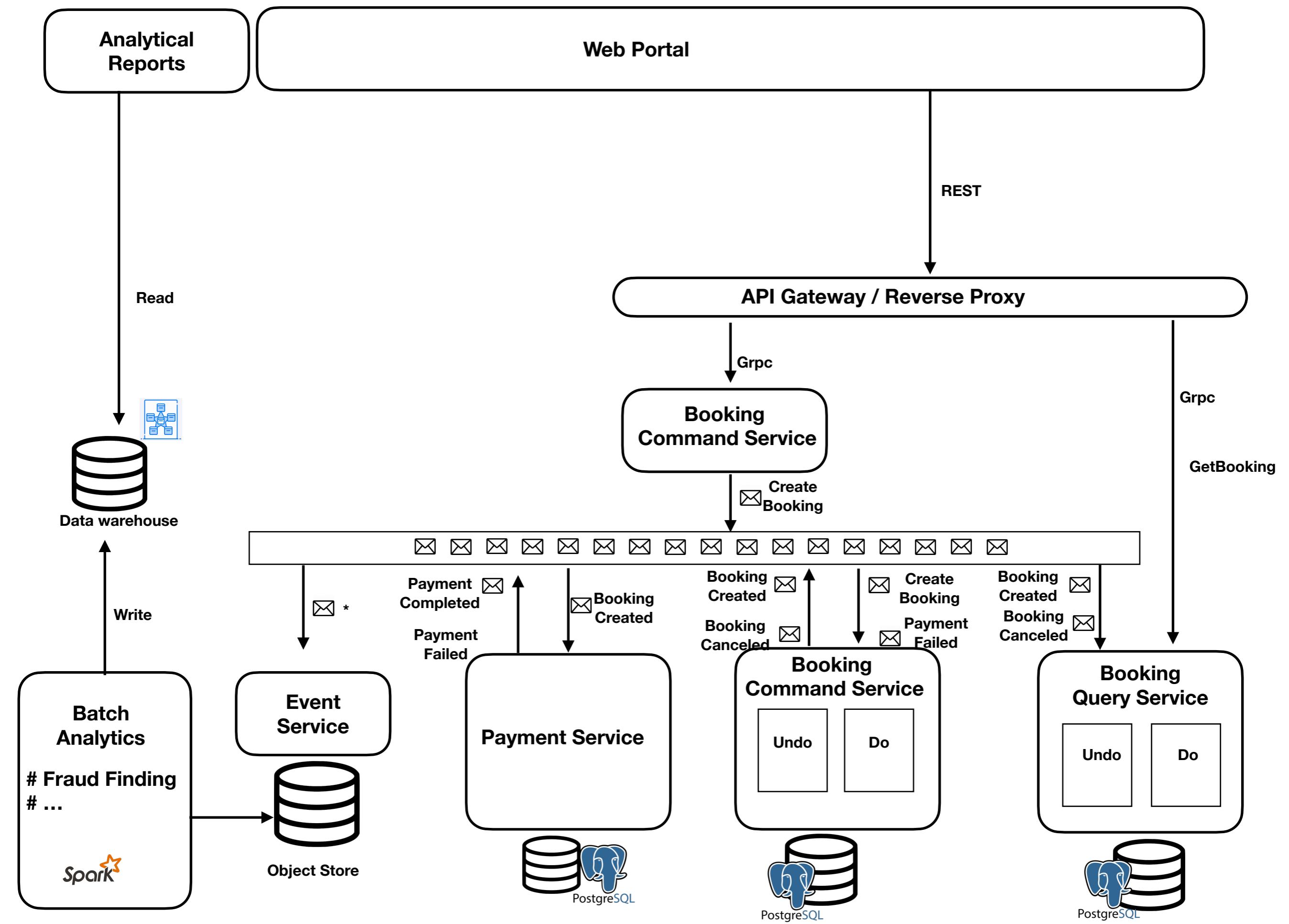






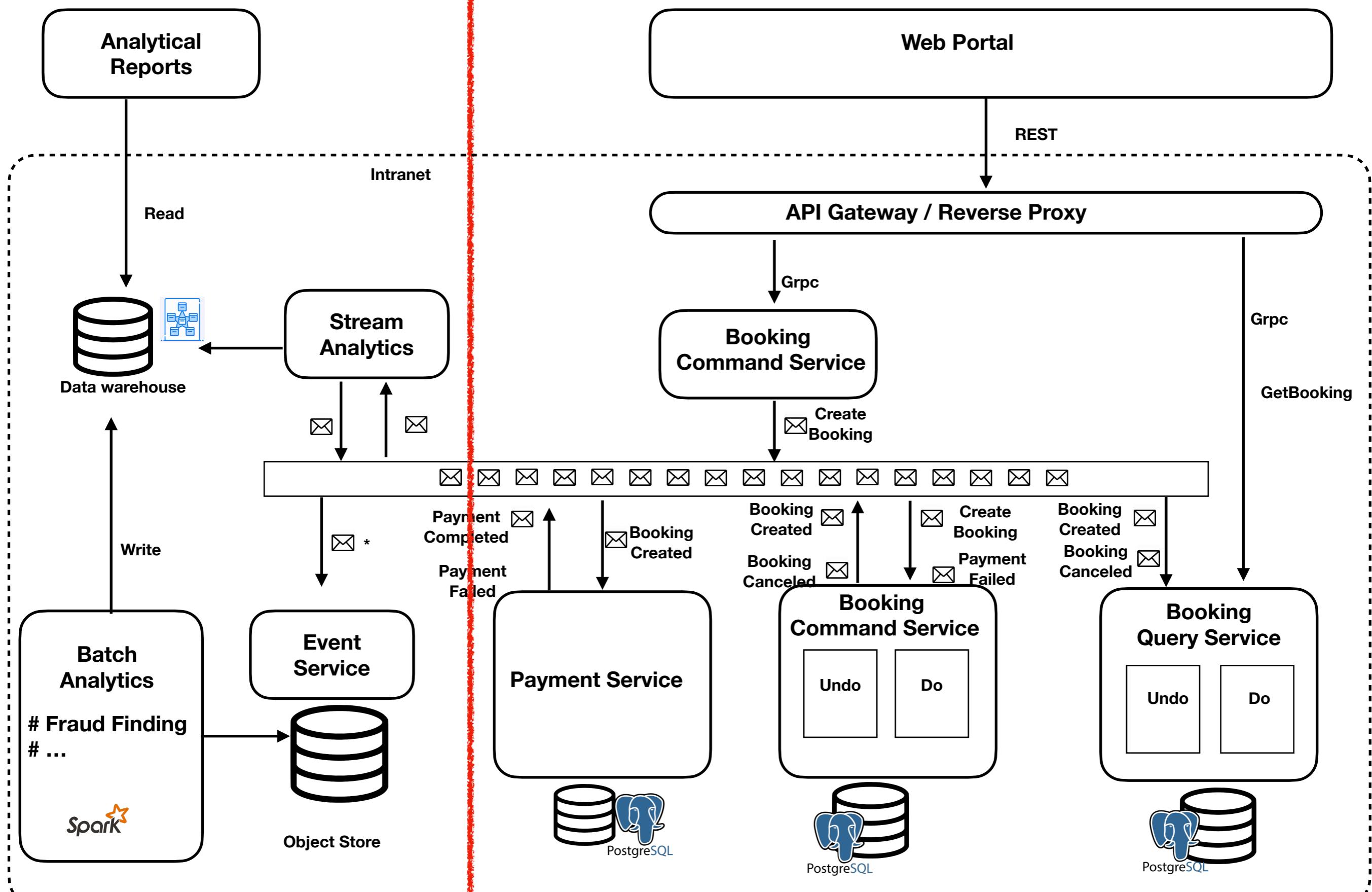
EDA
Event Sourcing
CQRS
SAGA pattern
- compensatable transaction





<<Manage>>

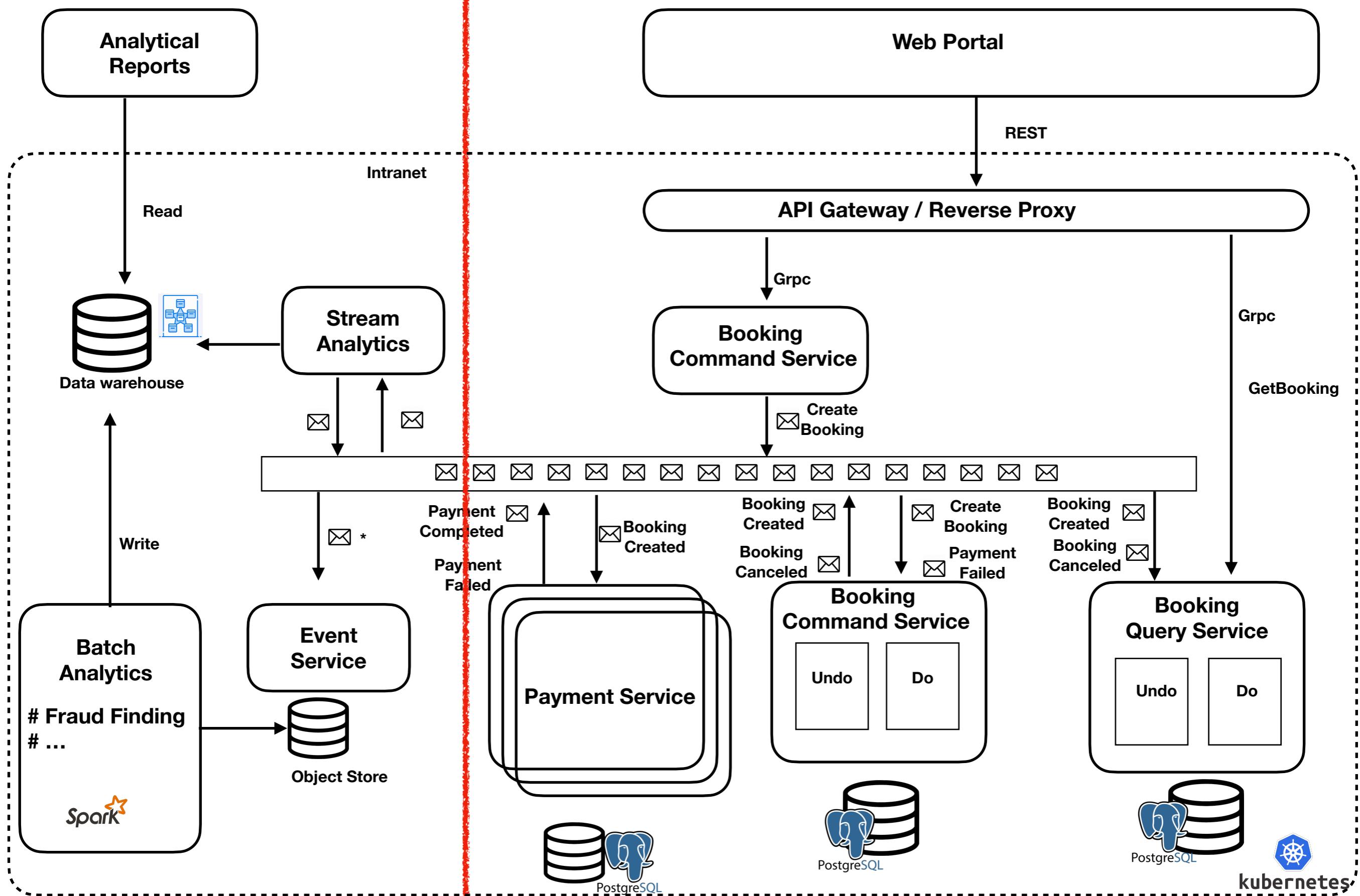
<<Run>>



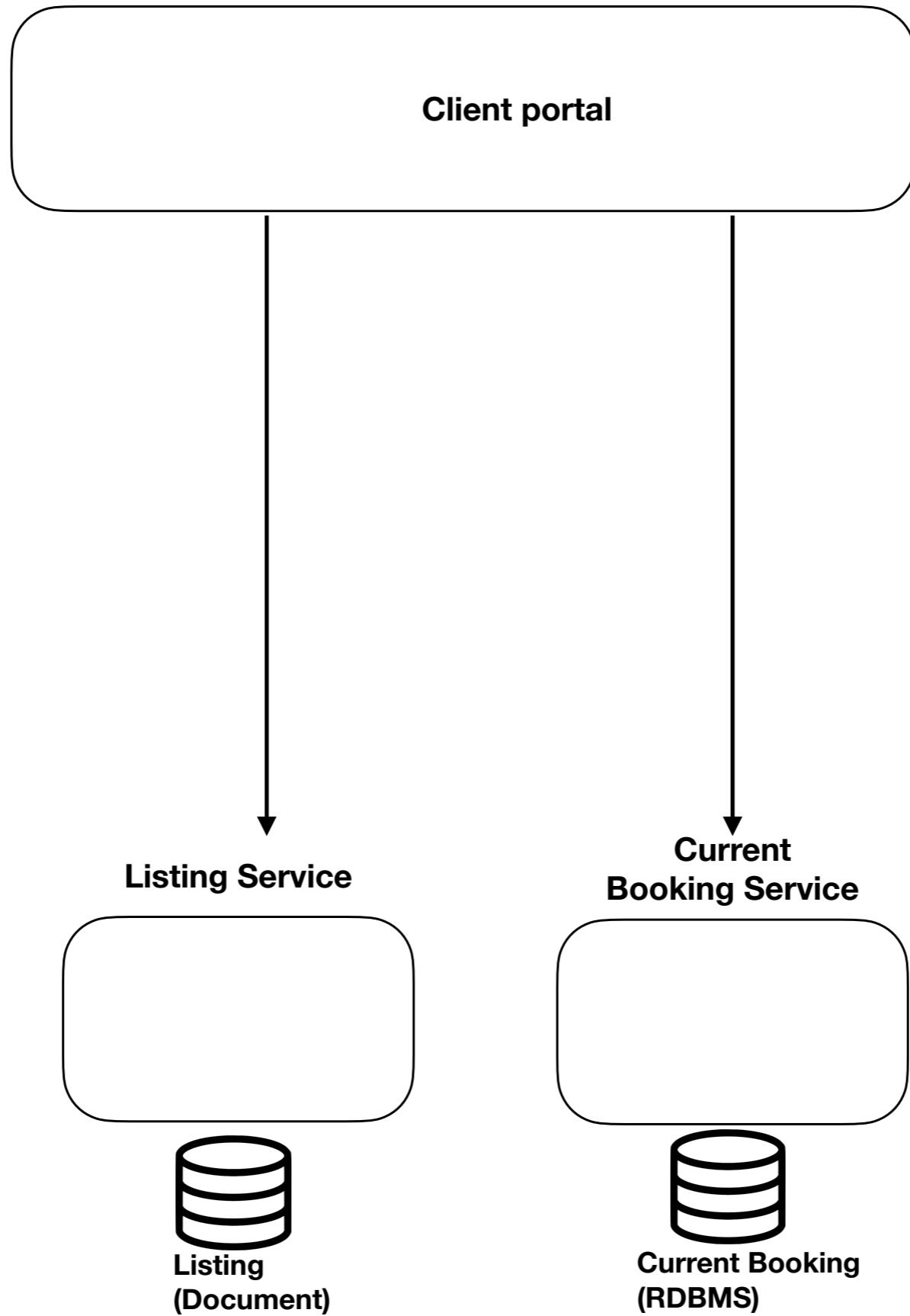
	Polling	Trigger	CDC	EDA
Scale	No	No	Yes	Yes
Causes extra load on db	Yes	Yes	No	No
Vendor locking	No	Yes	Yes	No
Low coupling	Yes	Yes	Yes	Yes

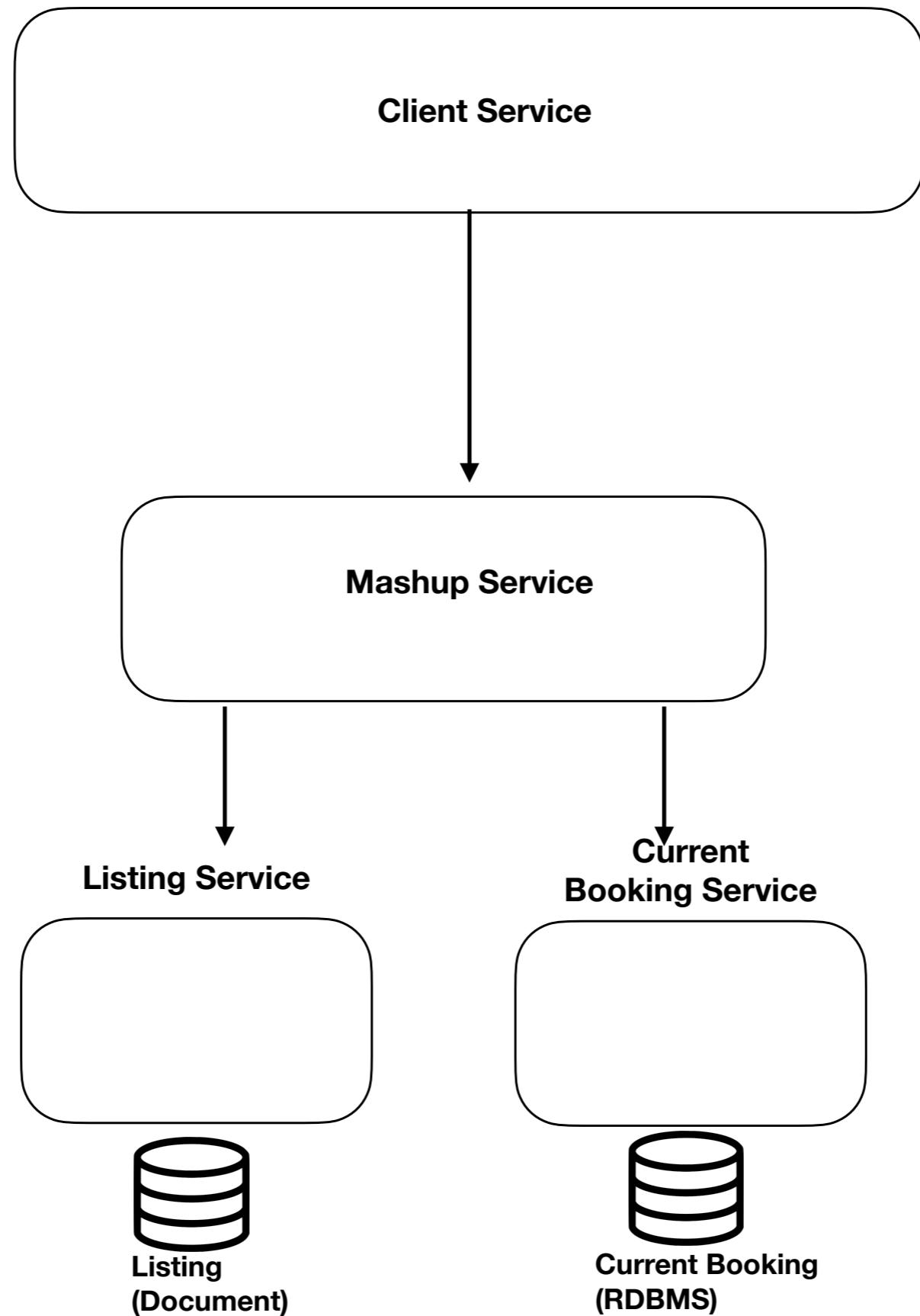
<<Manage>>

<<Run>>



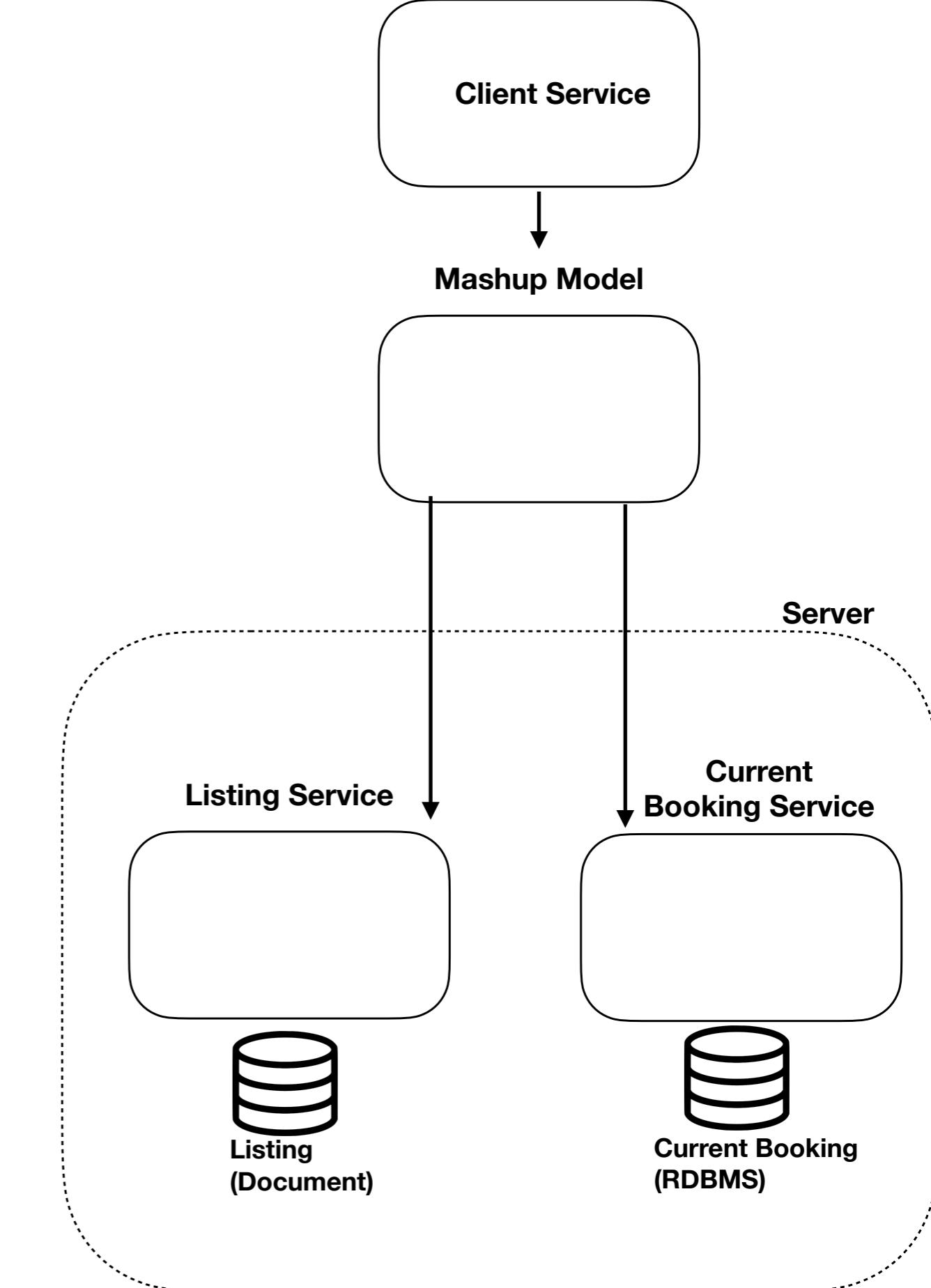
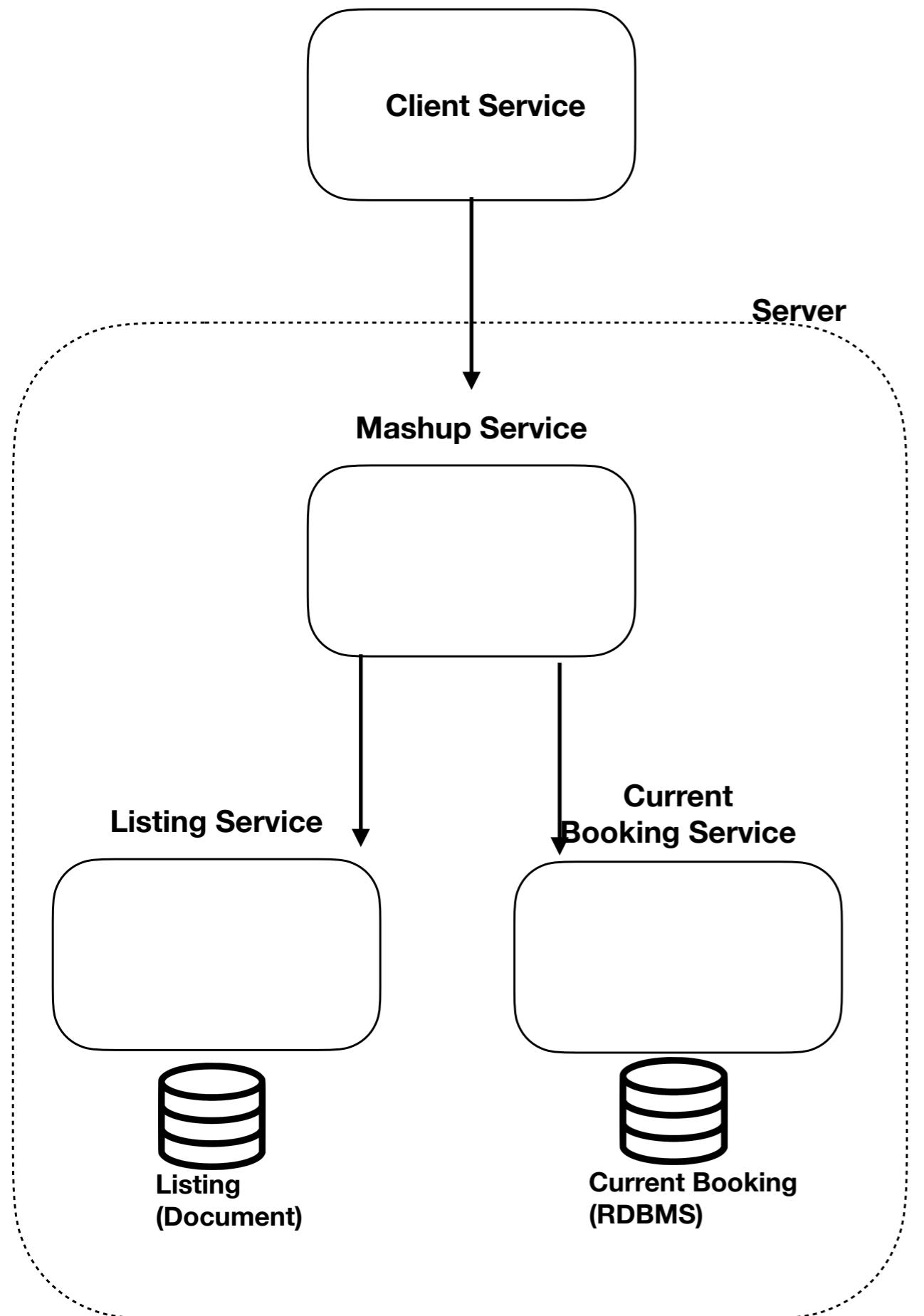
Distributed join





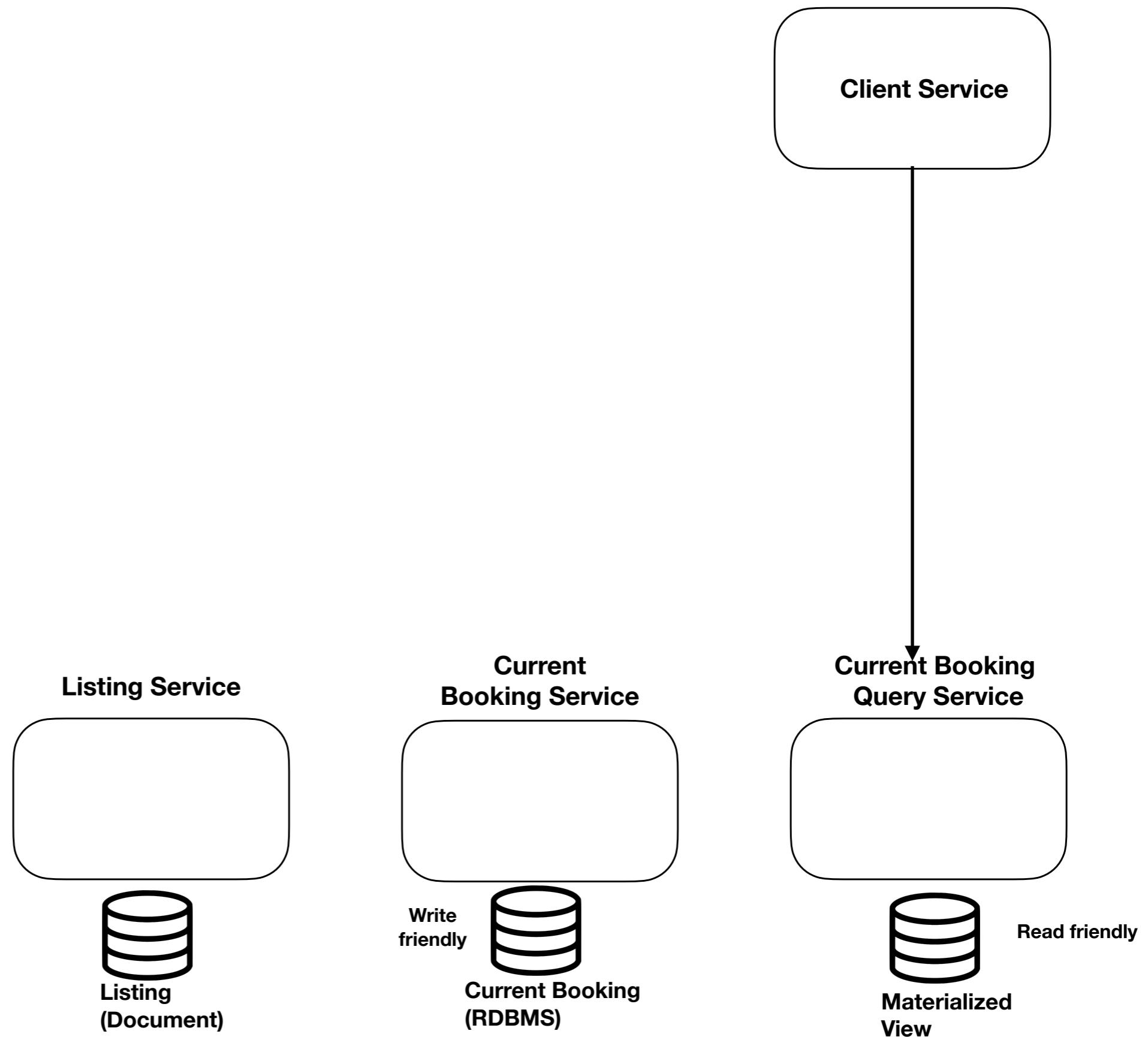
1

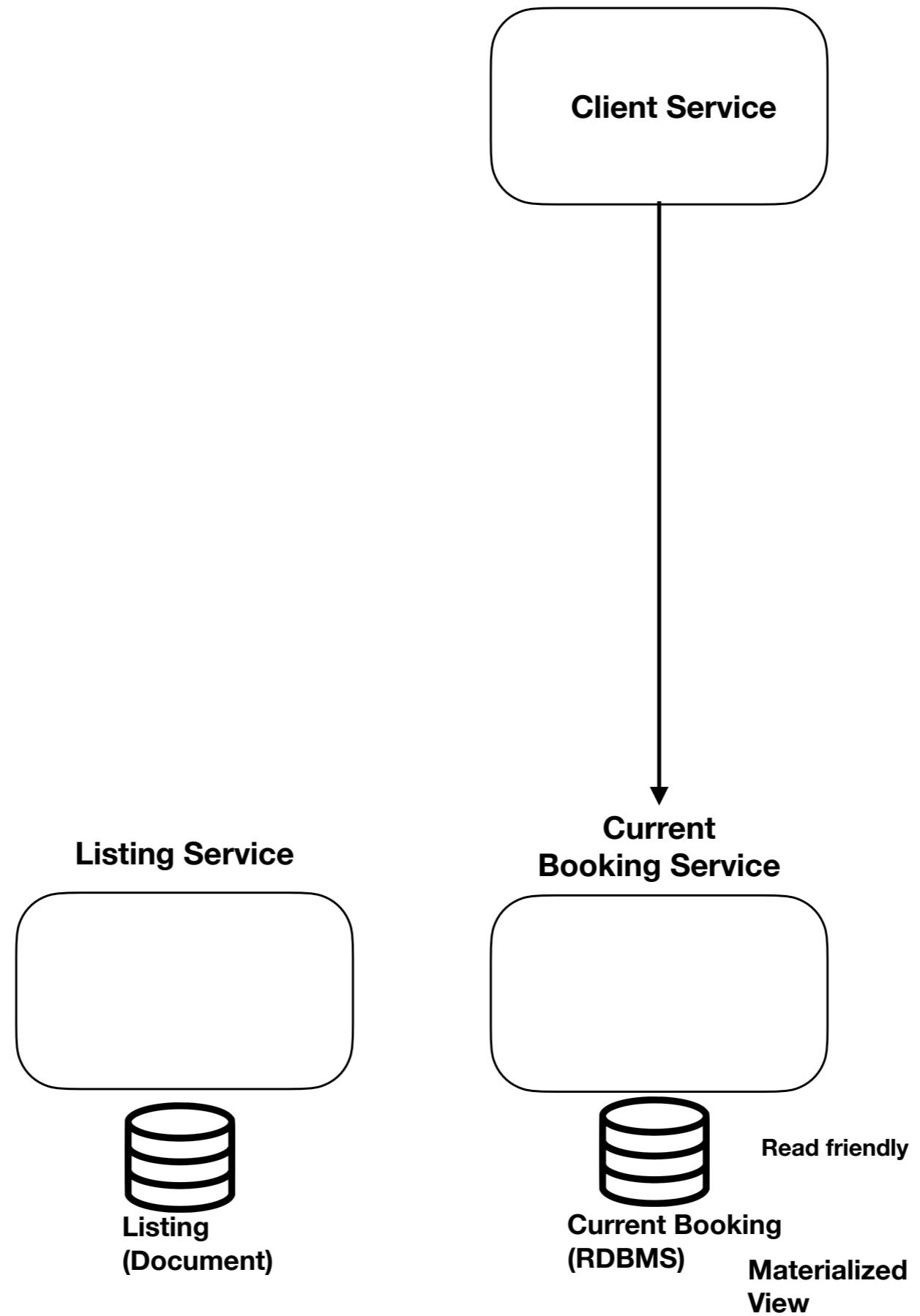
2

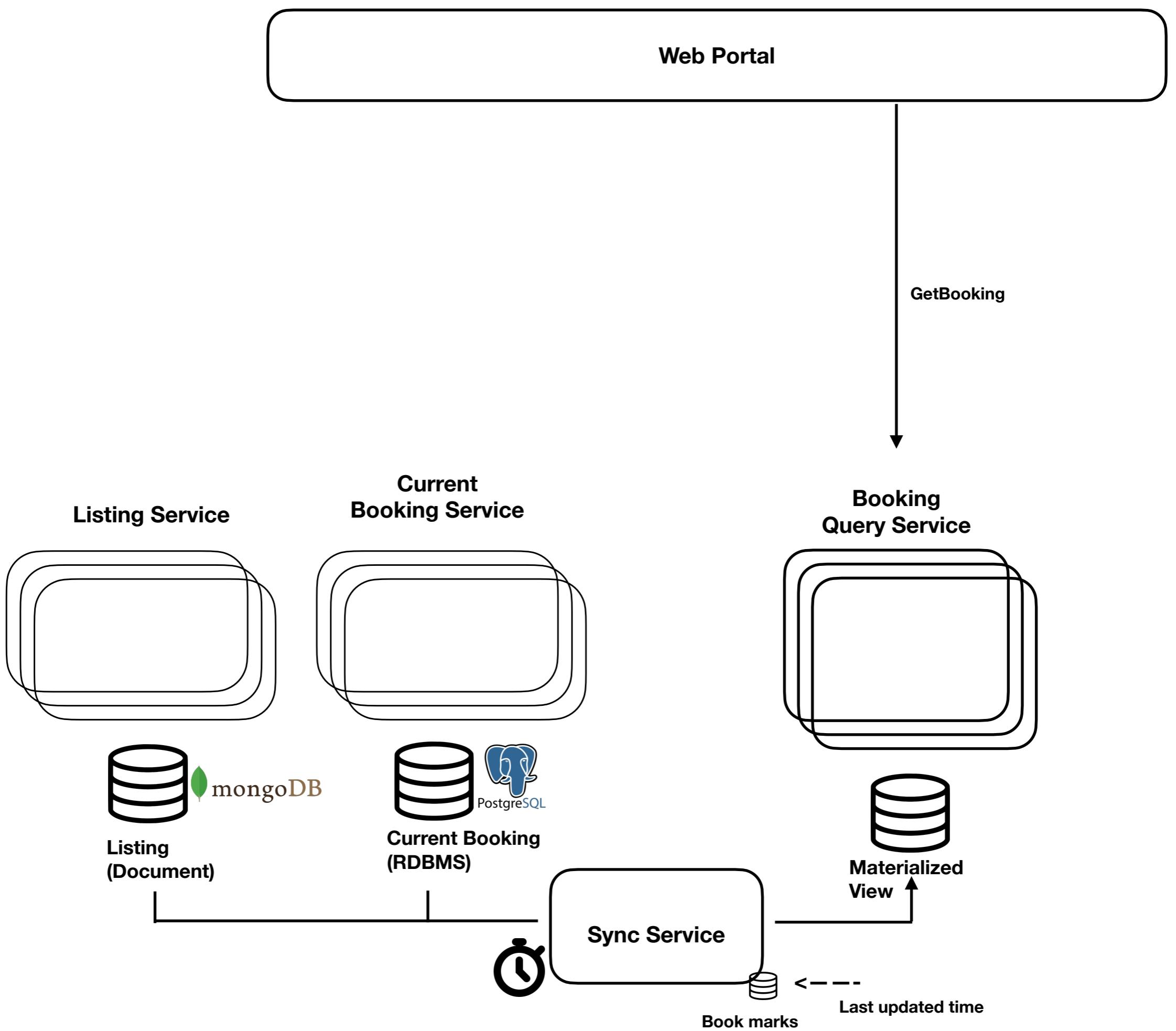


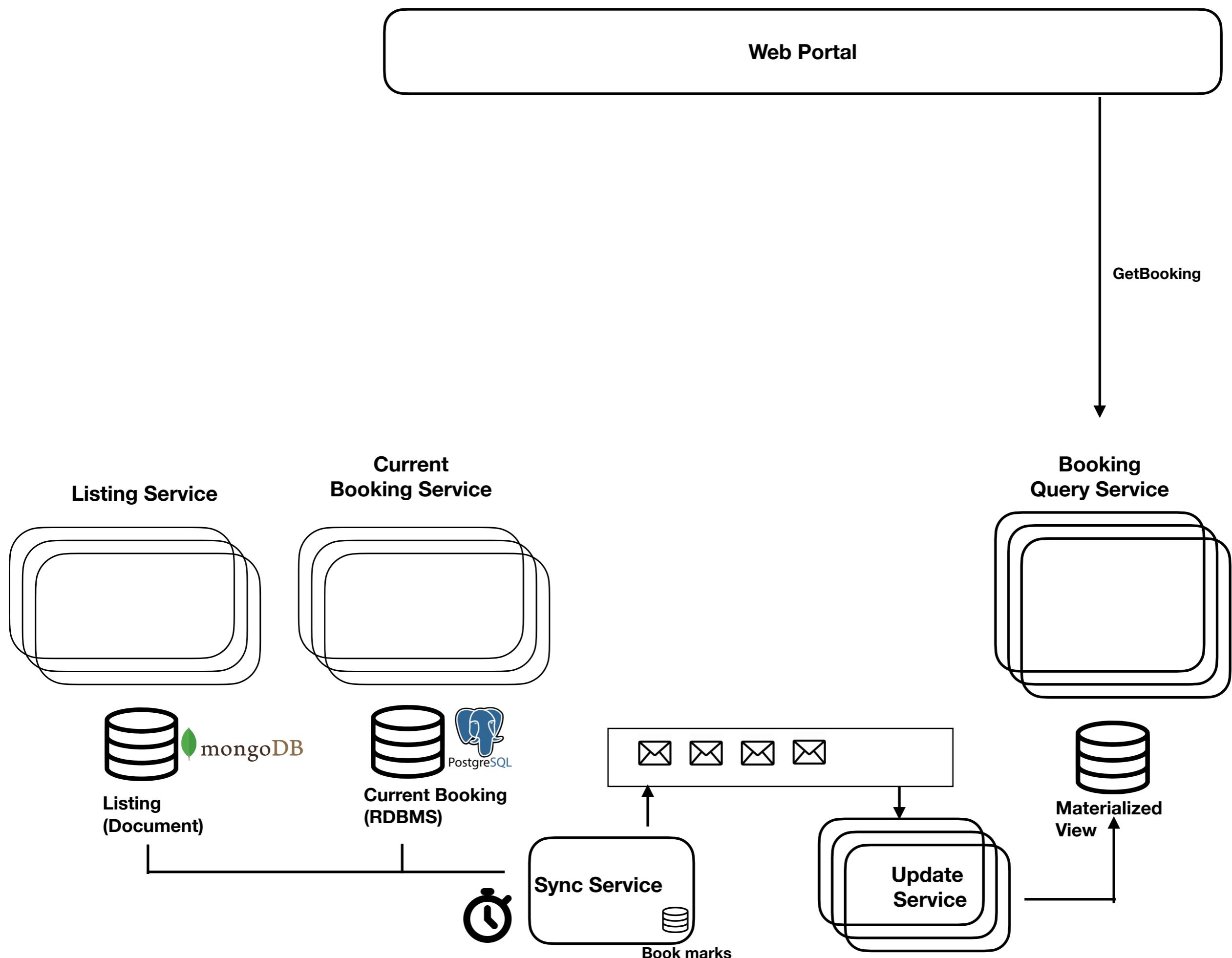
Perf

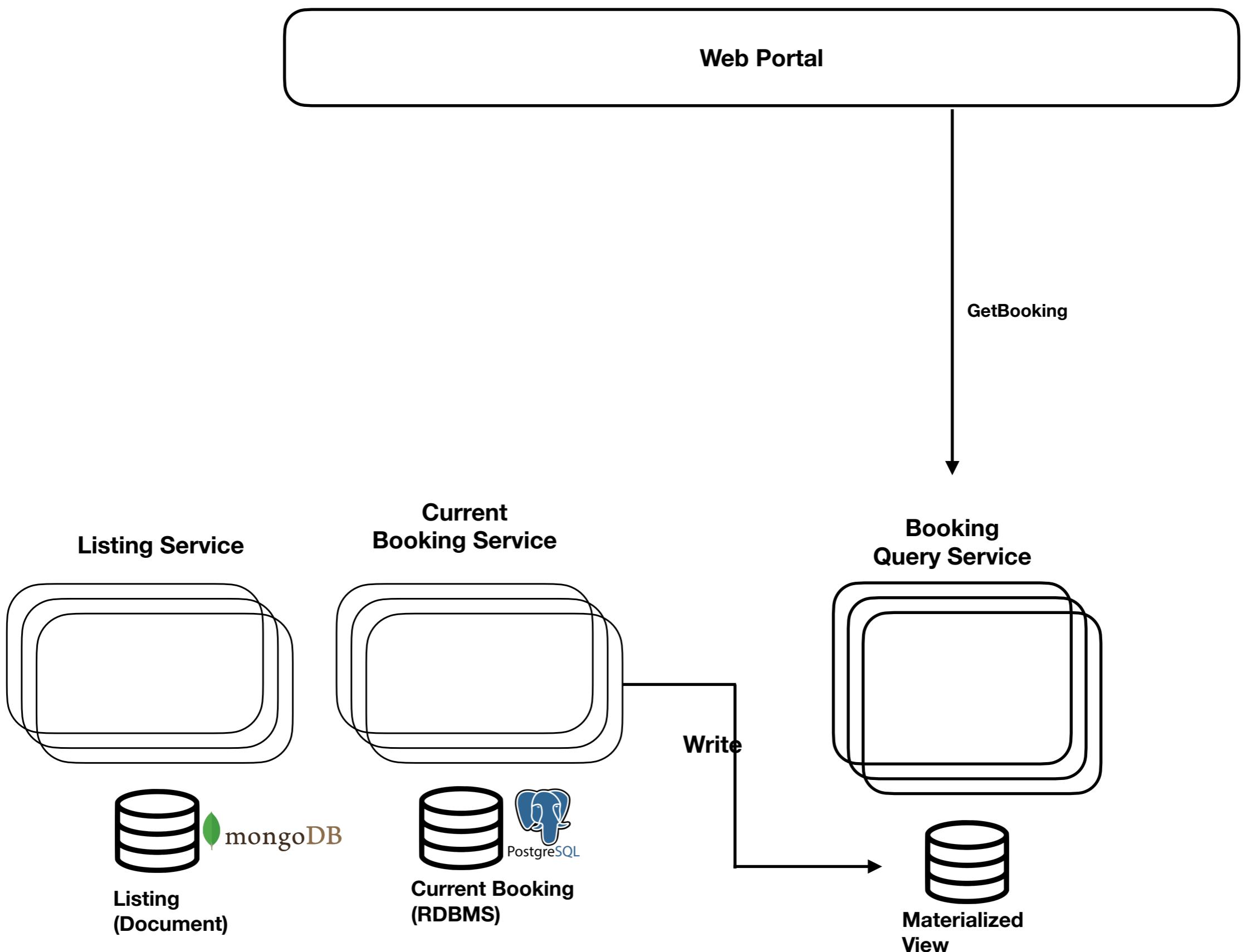
- $x + y \rightarrow$ 3 cpu cycles
- Fun call $\rightarrow \sim 10$ cpu cycles
- Create thread $\rightarrow 200,000$ cpu cycles (kernel)
- Remote fun call $\rightarrow 10,00,000$ cpu cycles (IO)

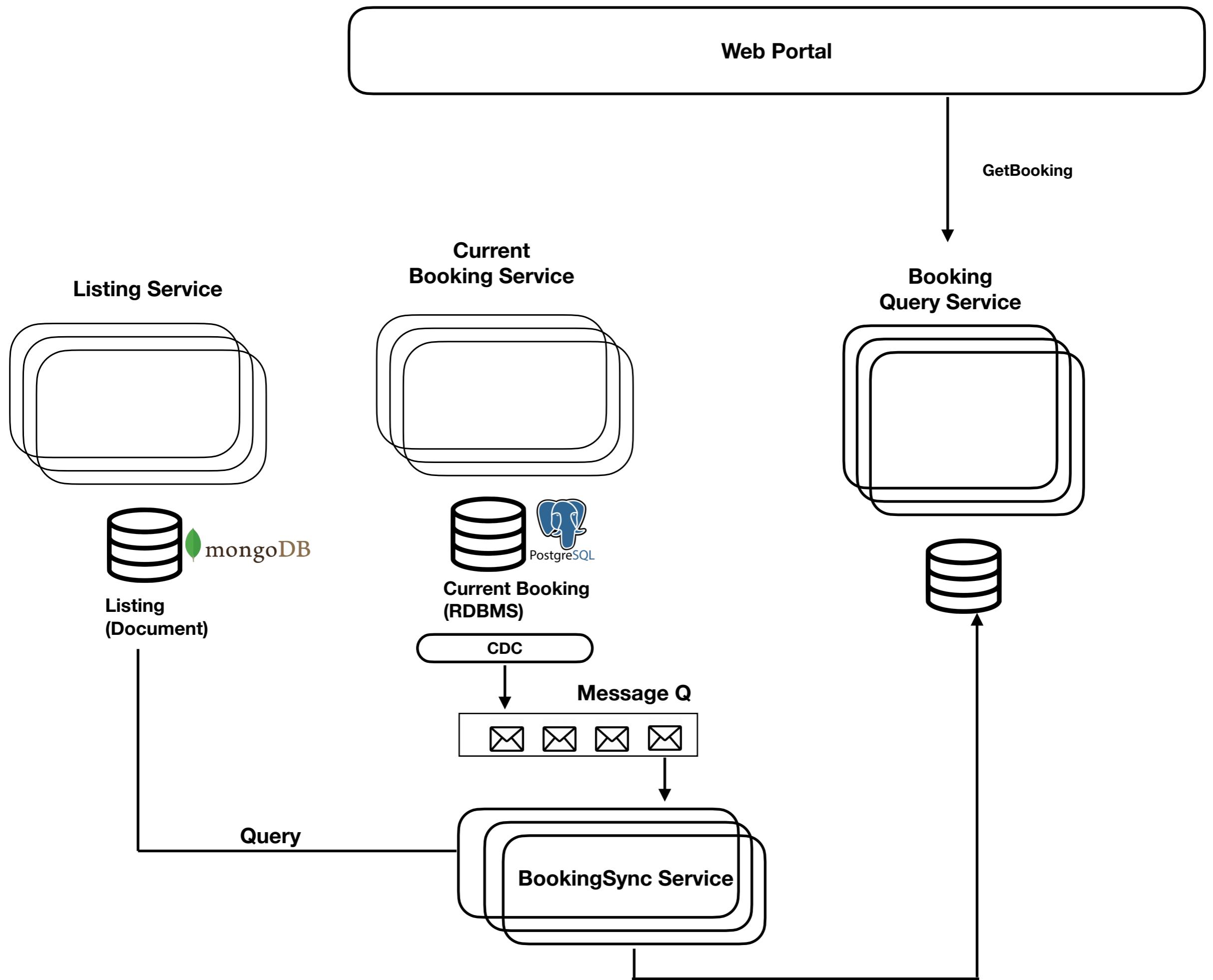






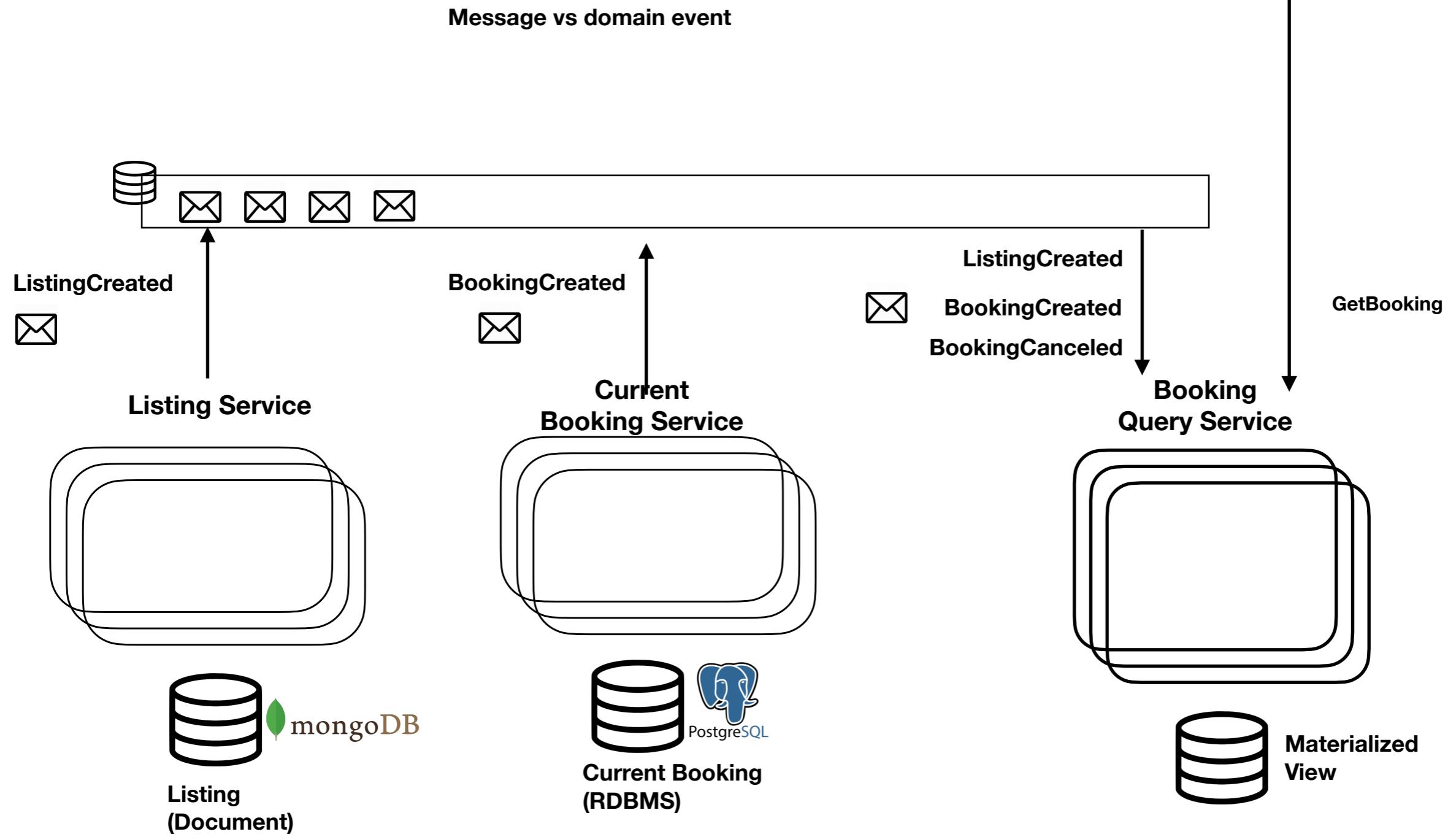


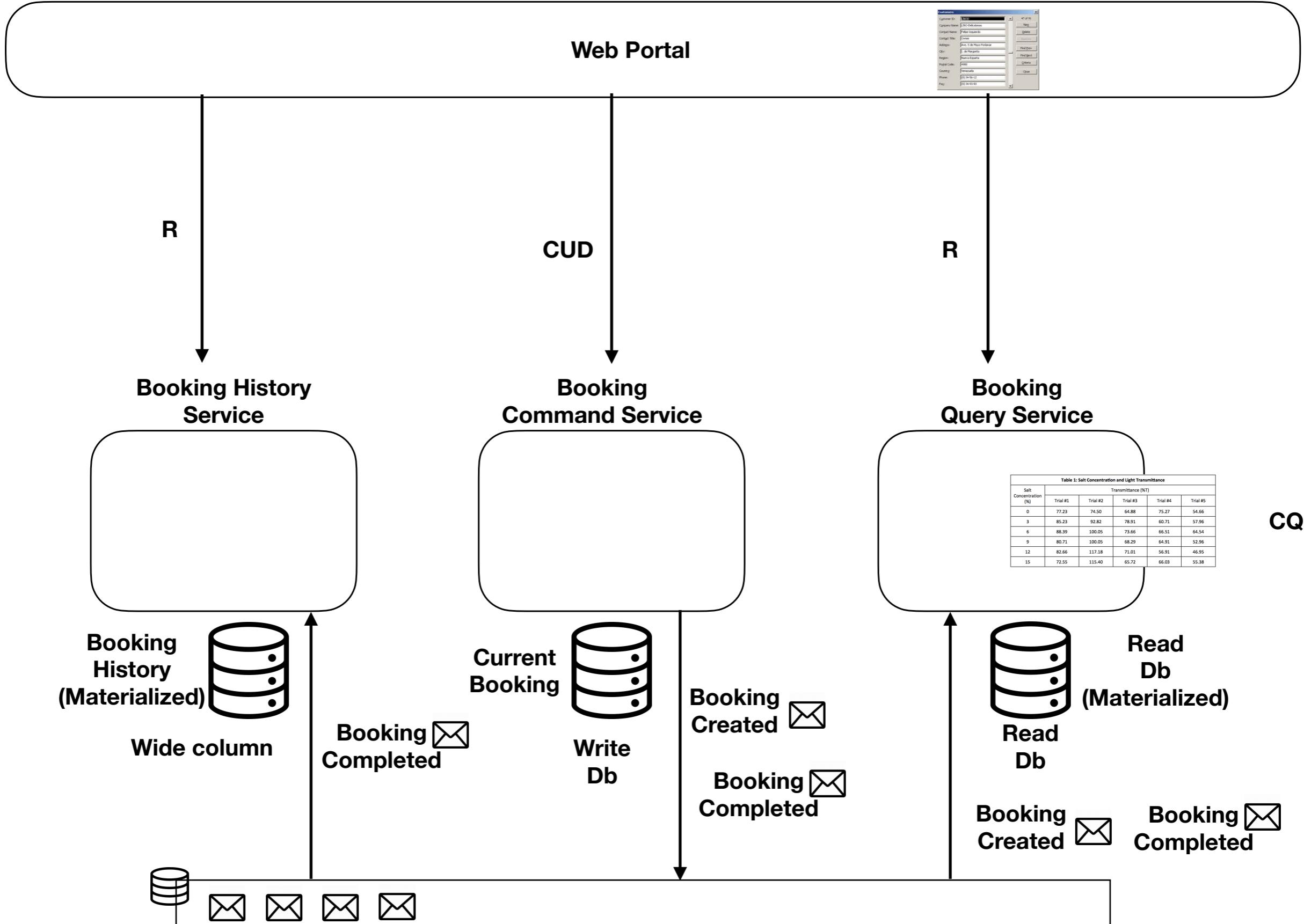


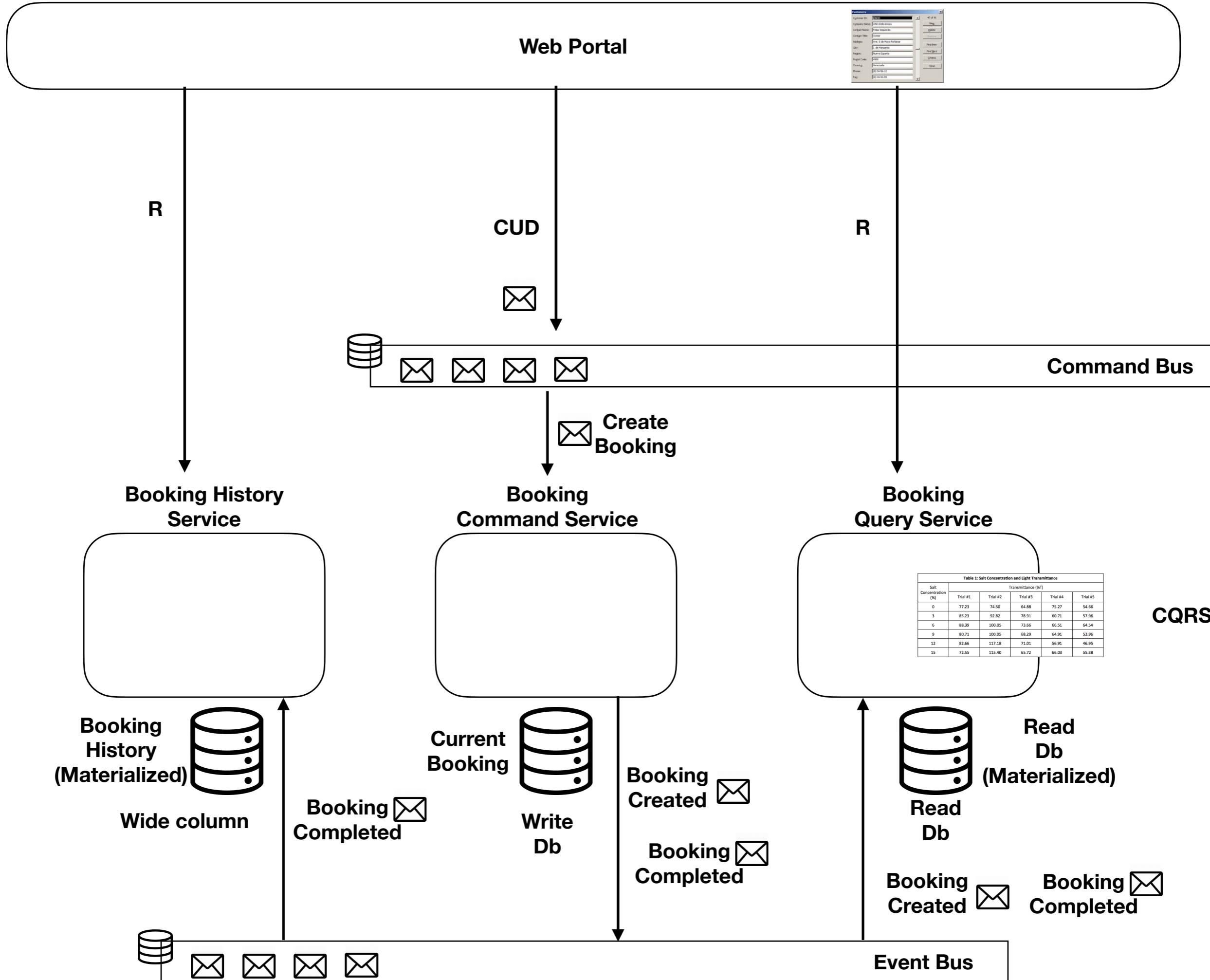


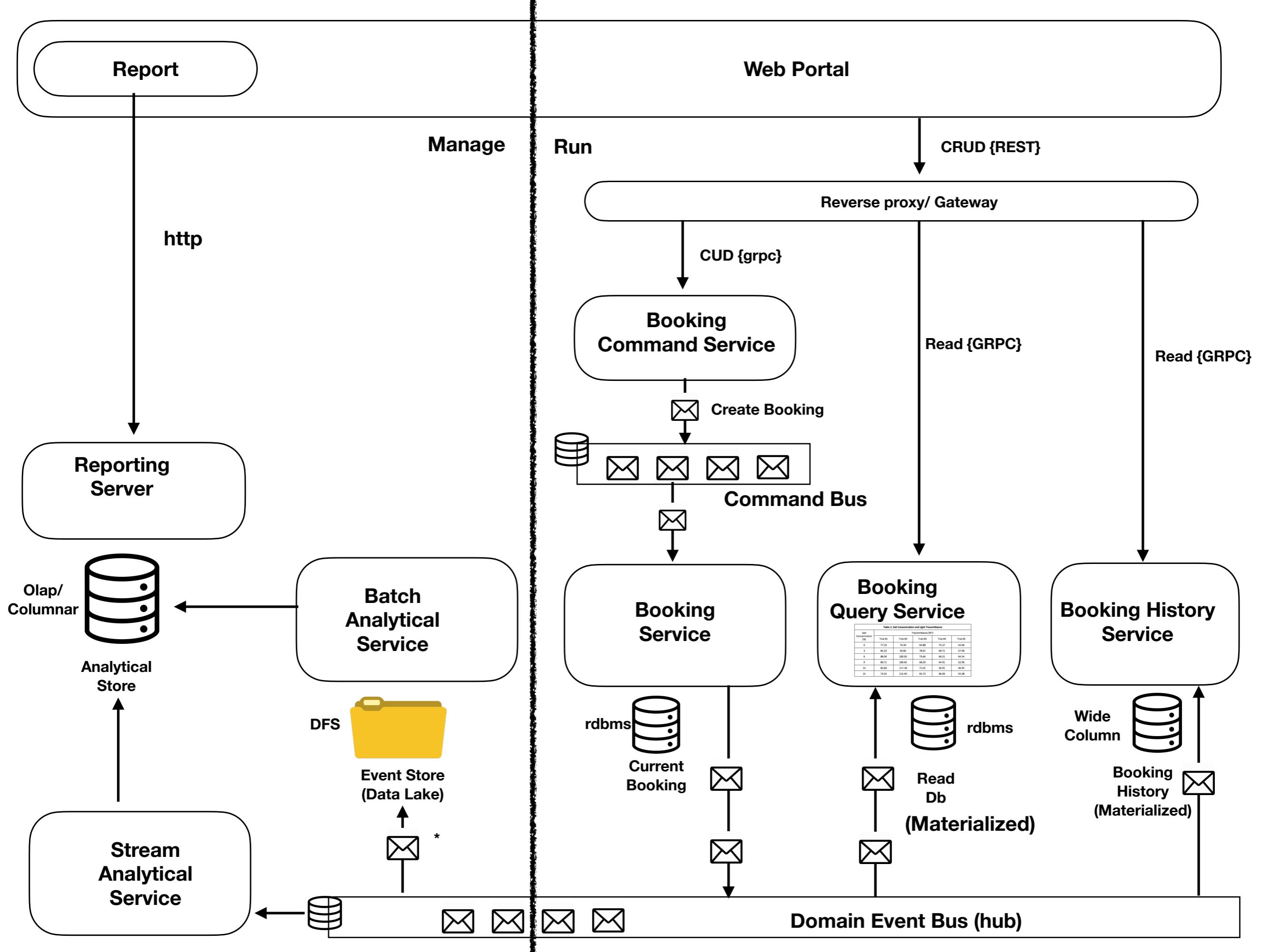
EDA
Domain Events
Event Sourcing

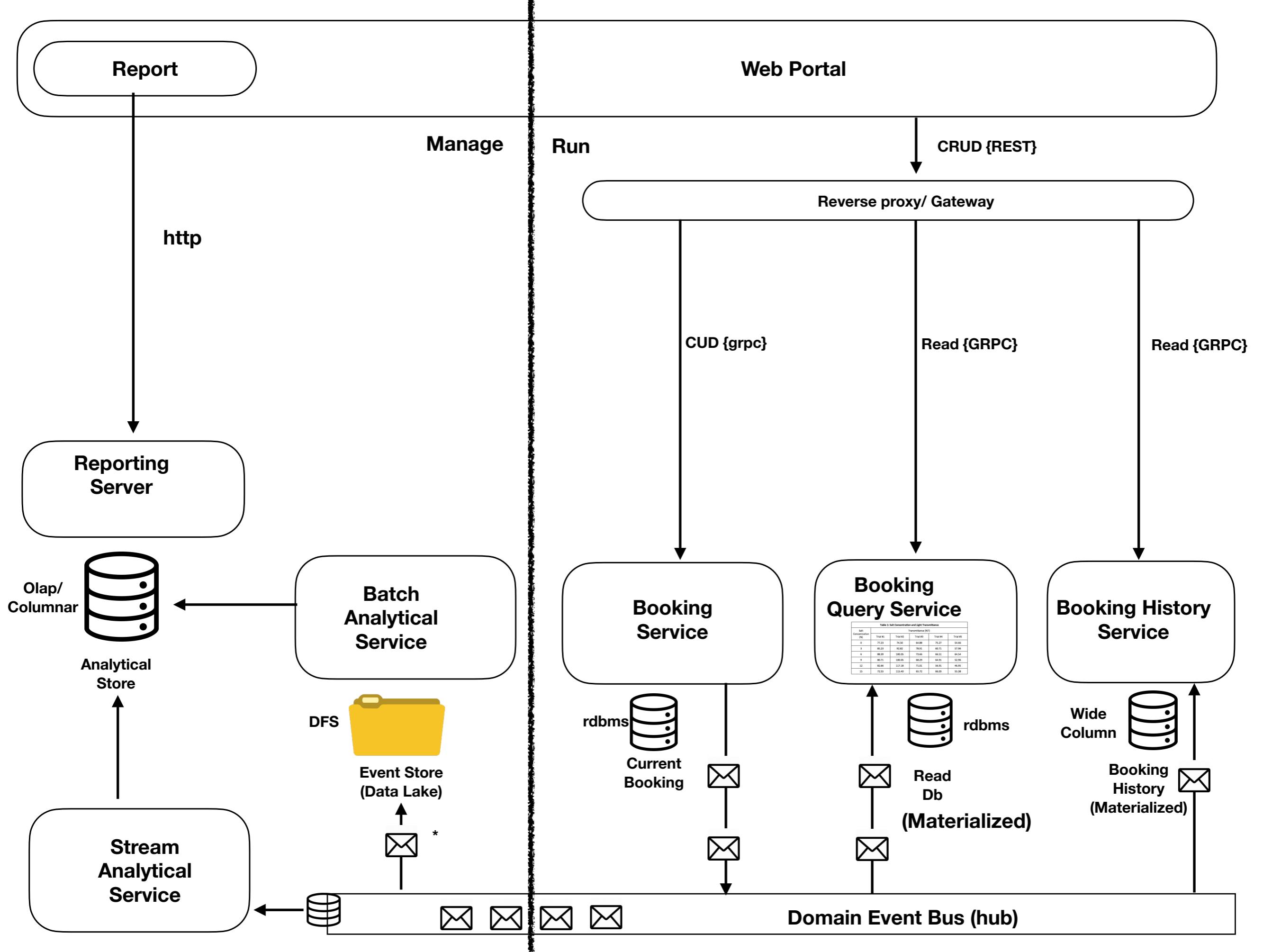
Web Portal

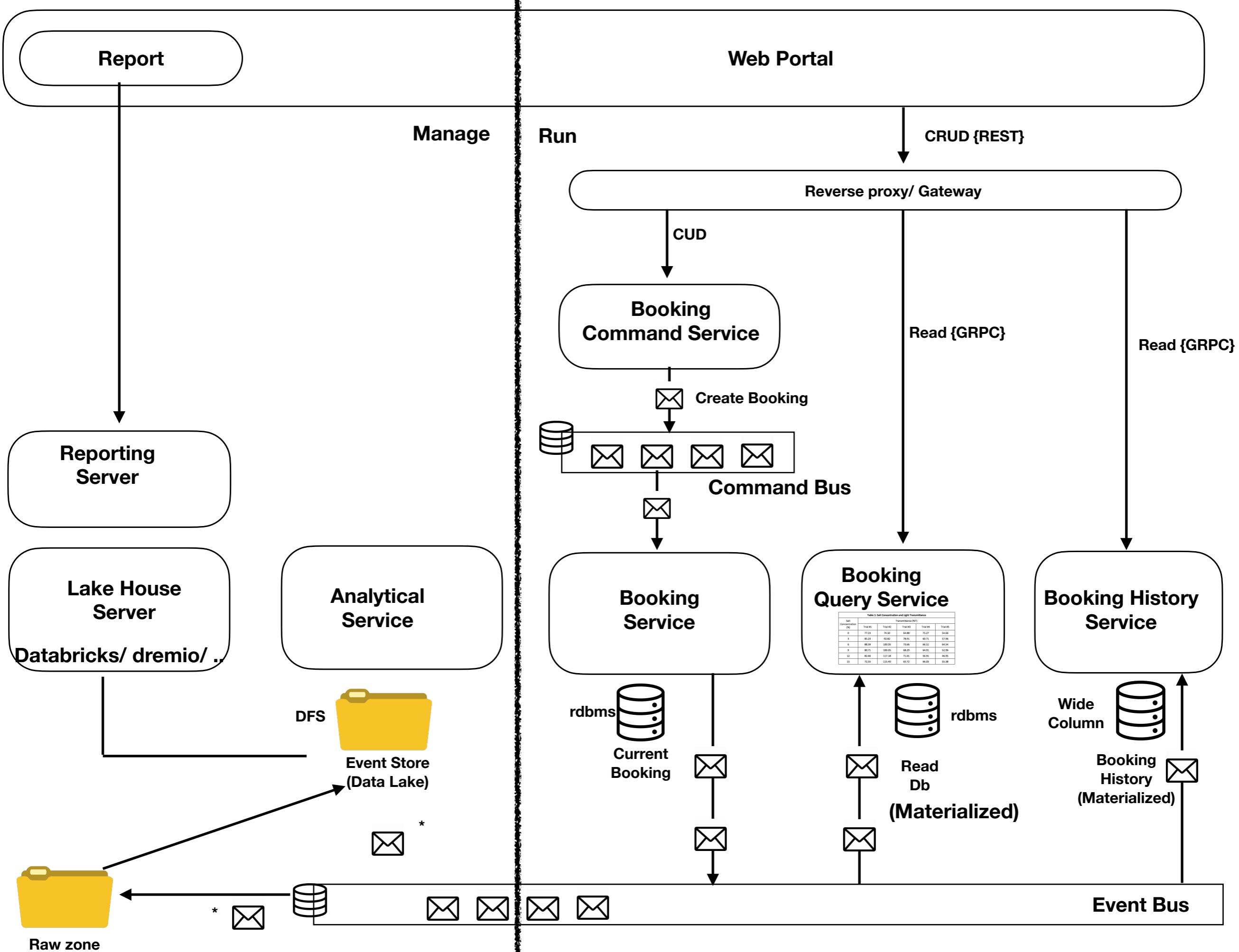


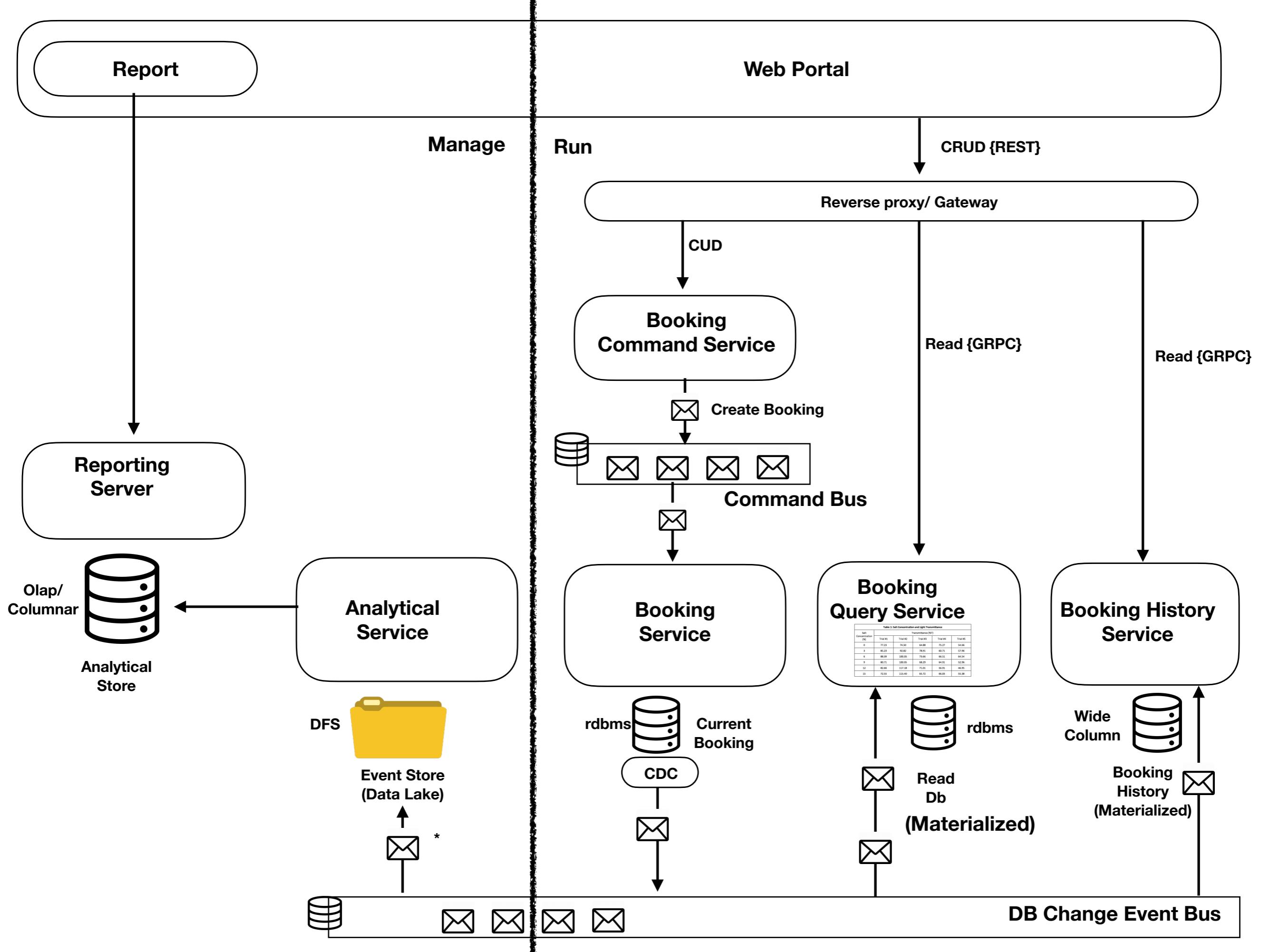


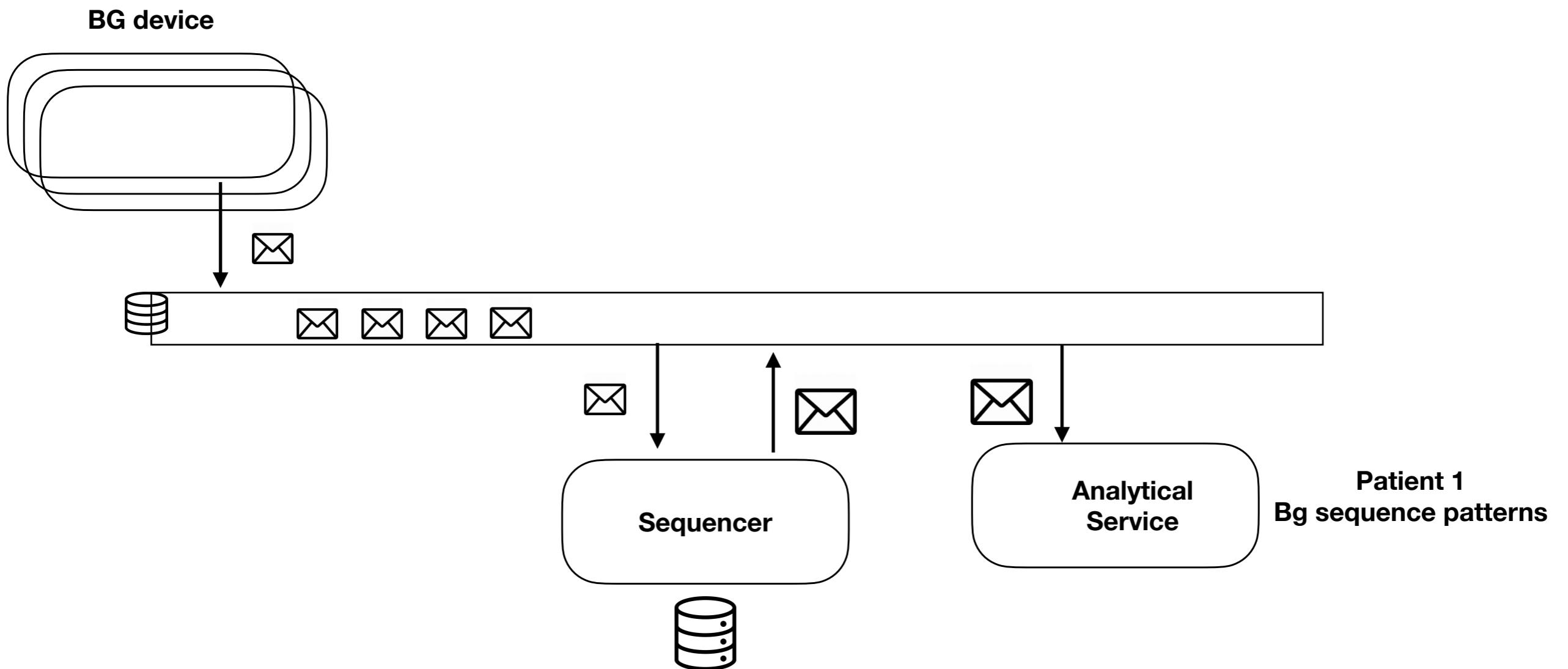


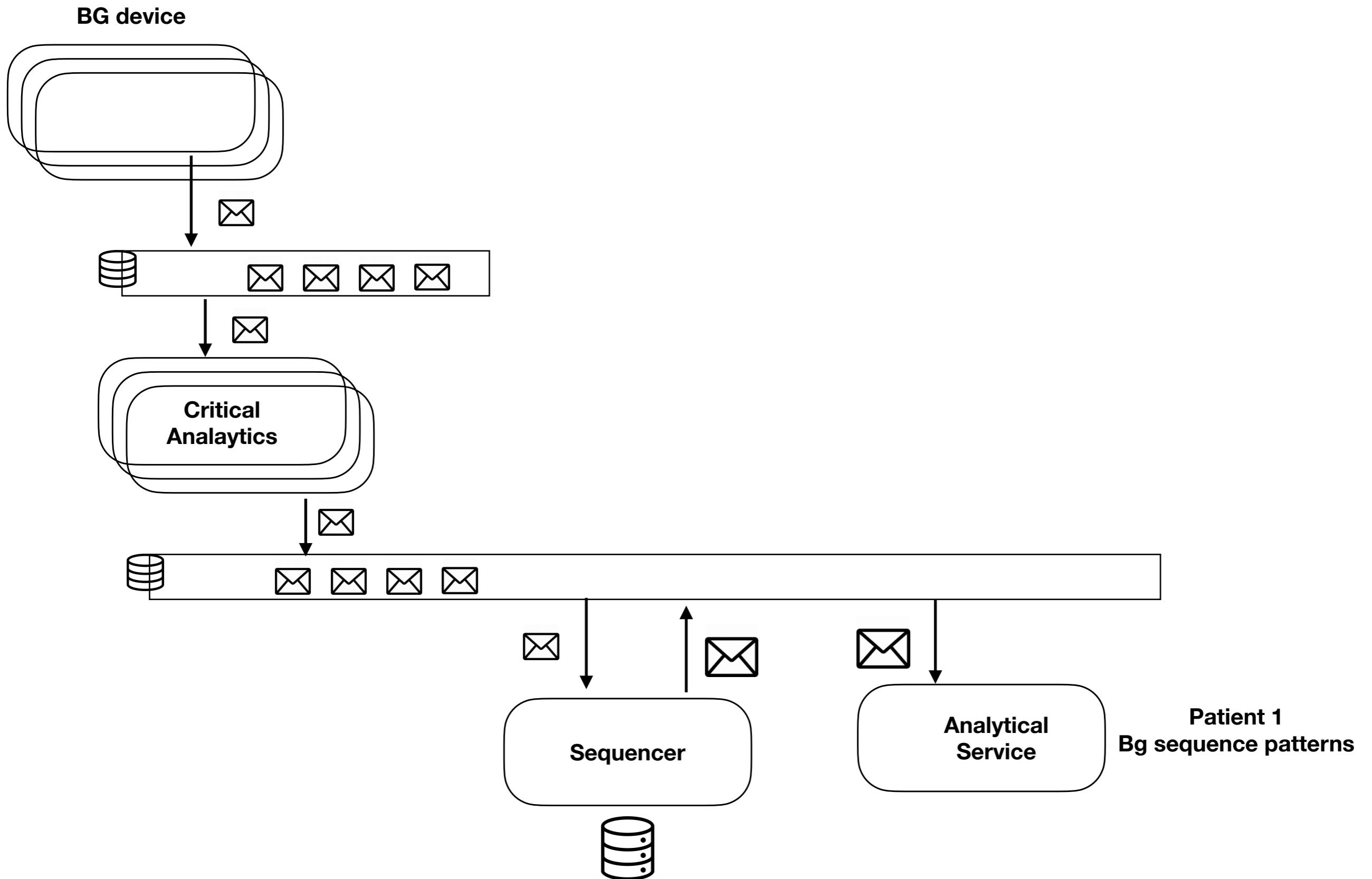




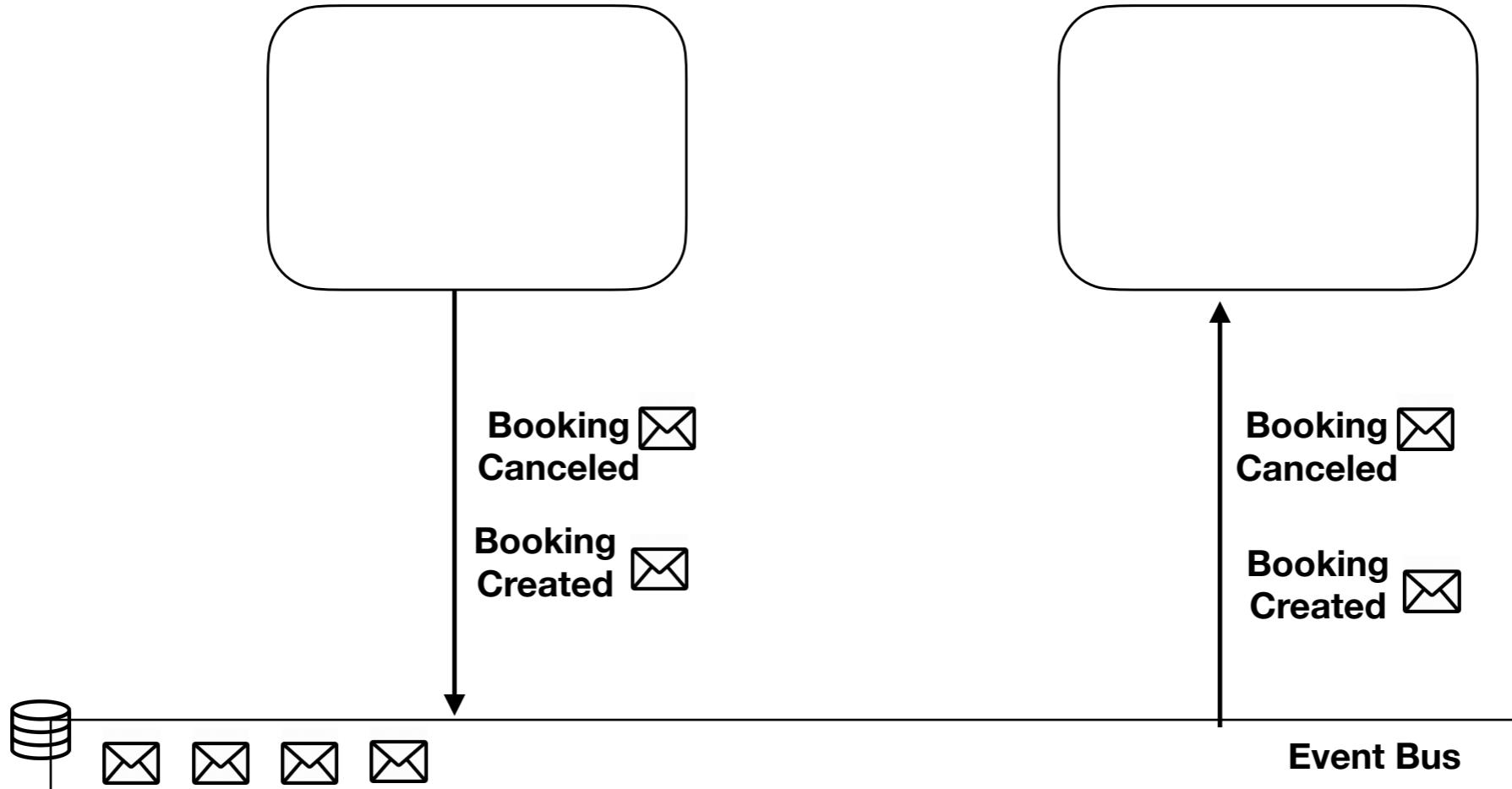


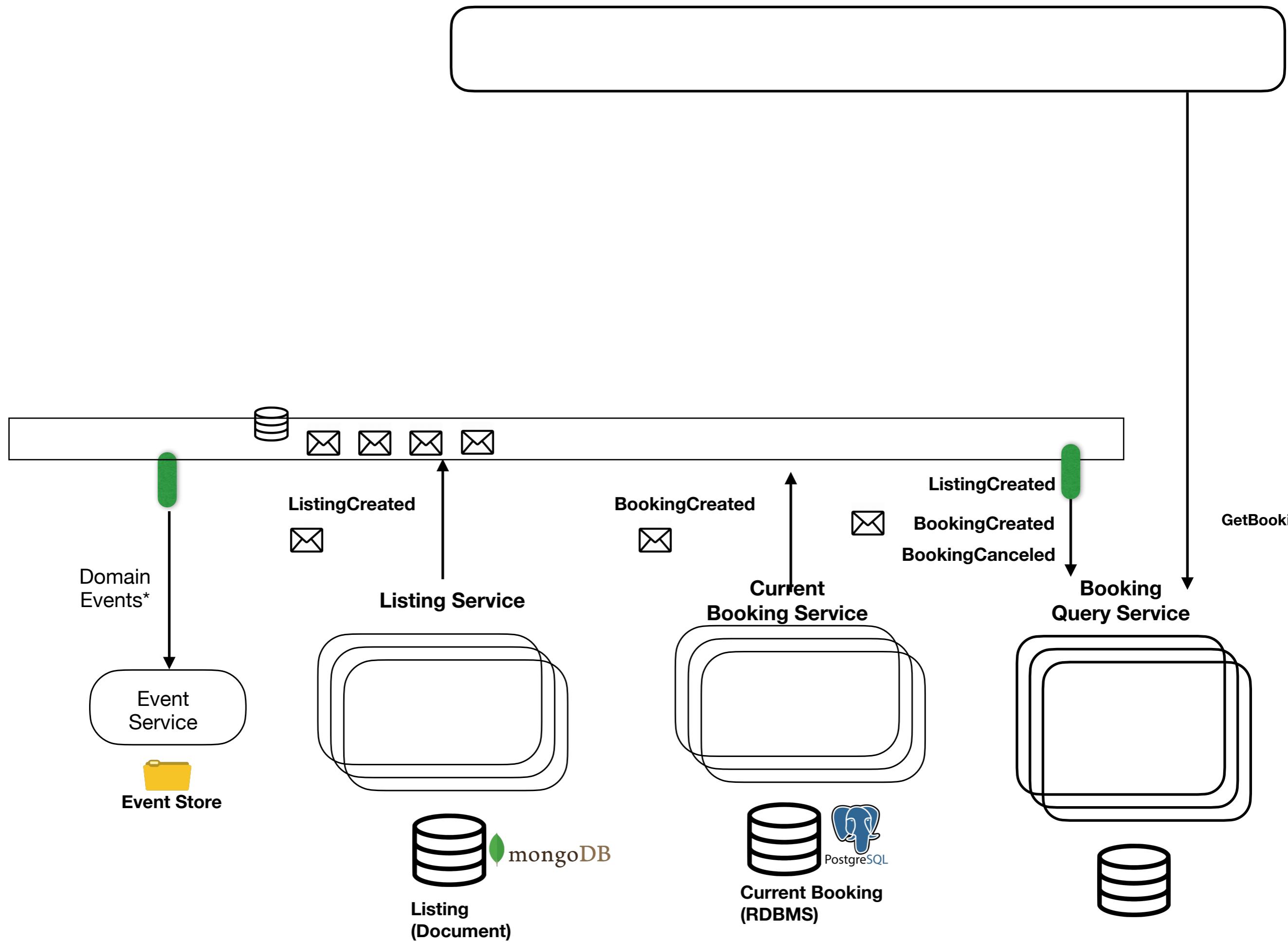






- EDA
- Domain Events
- Event sourcing (e1,e2,e3,...)
- CQRS
- Materialized views
- CDC

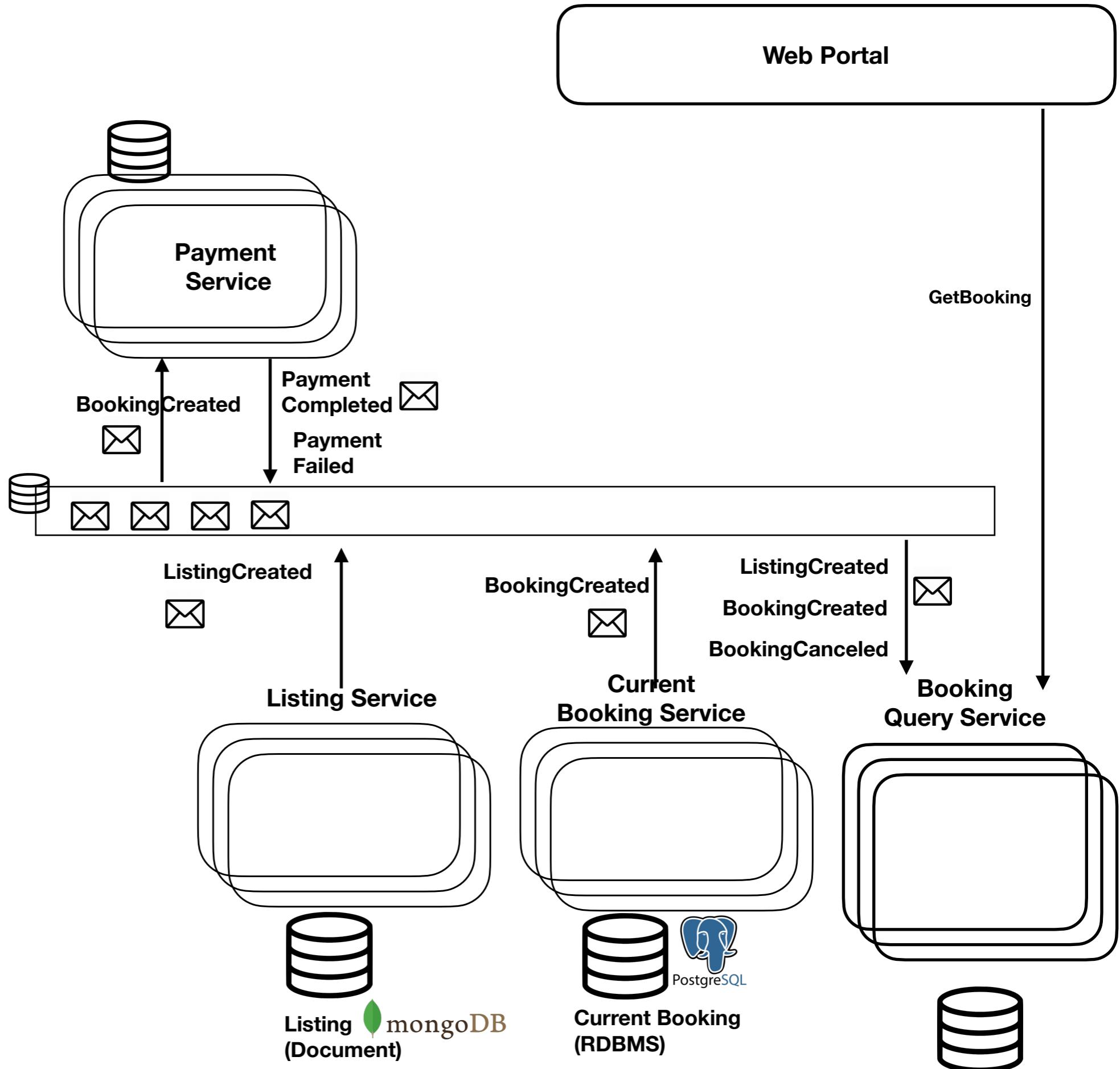




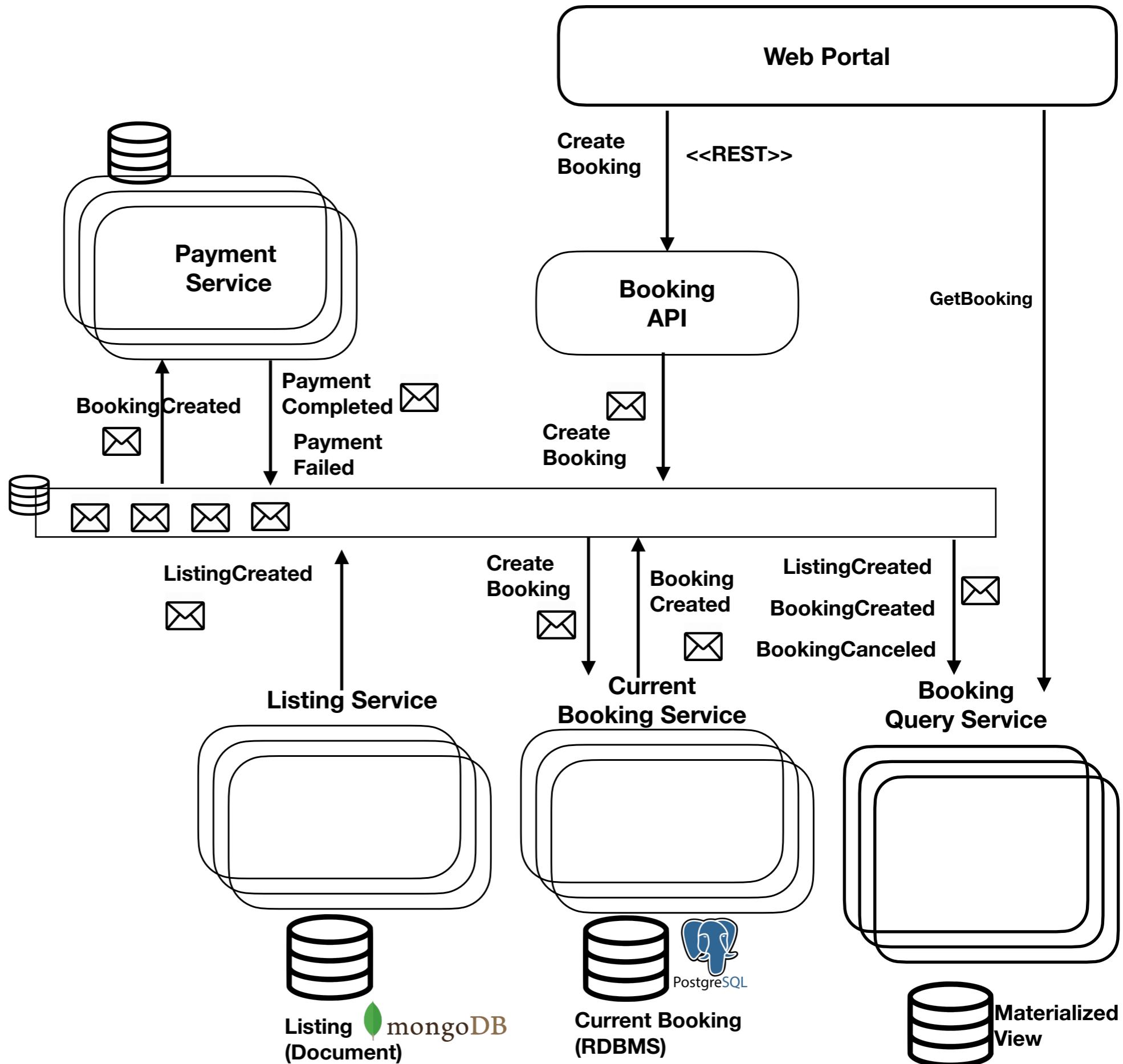
CDC vs domain event

Distributed transaction

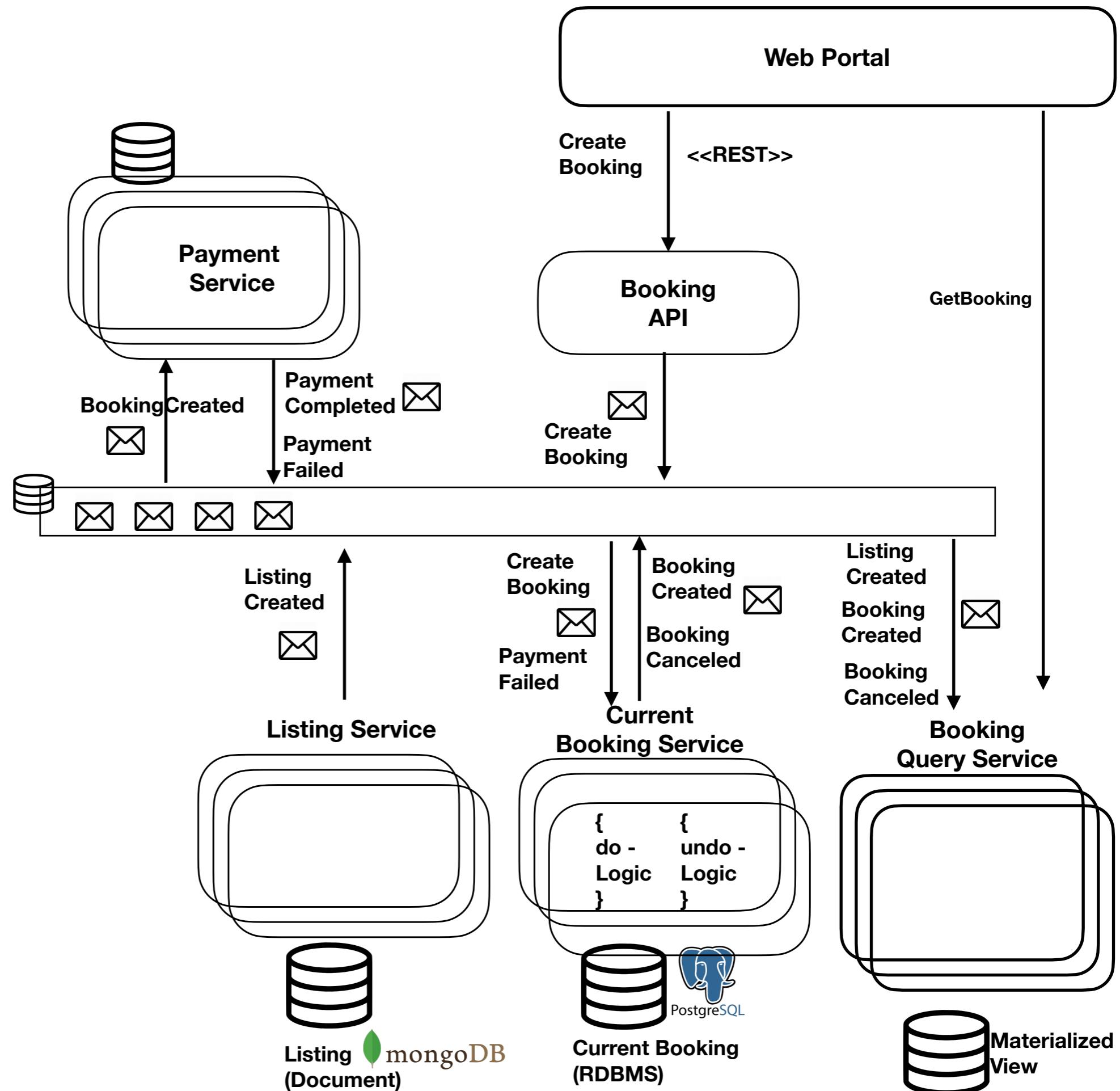
```
# Domain Command  
# Domain Events  
# Domain Exception
```

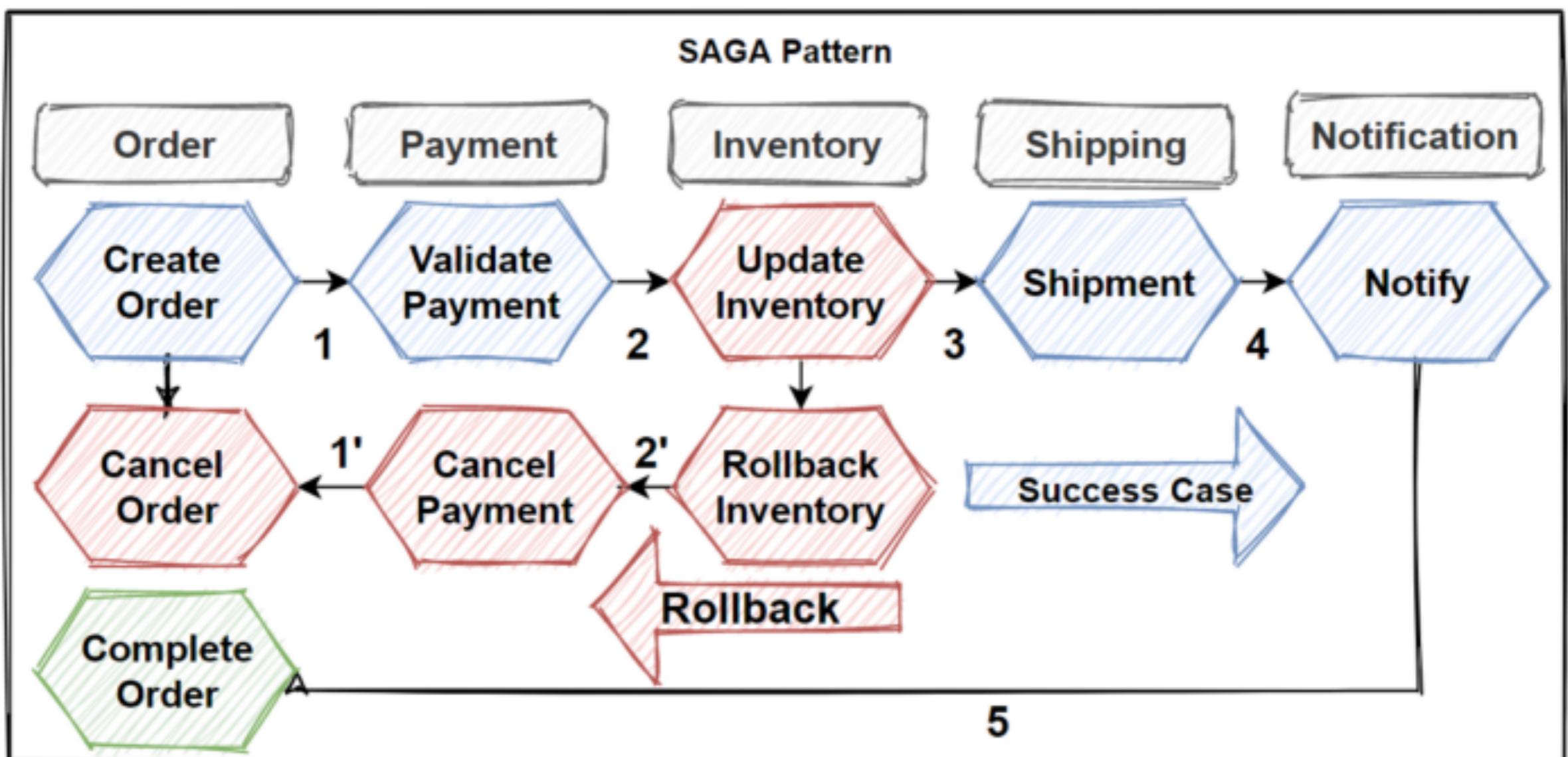


```
# Domain Command  
# Domain Events  
# Domain Exception
```

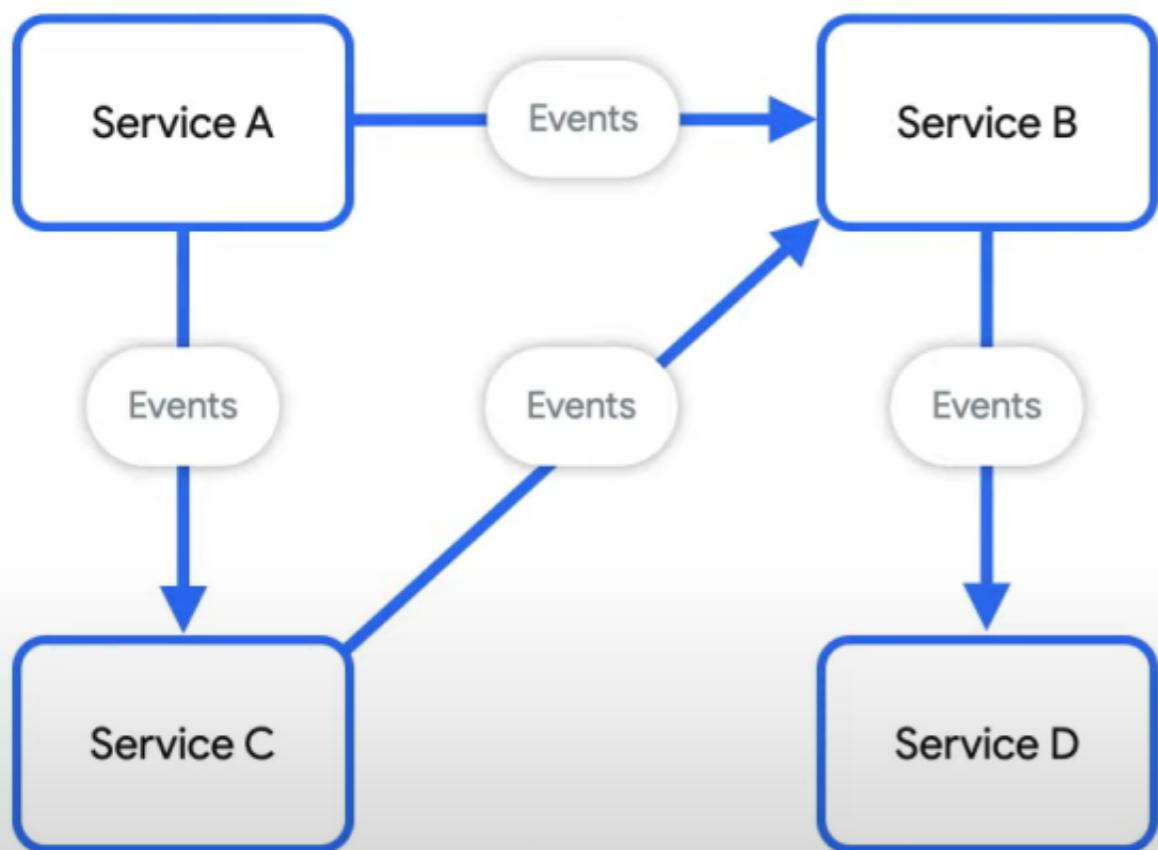


```
# Domain Command  
# Domain Events  
# Domain Exception  
# Compensatable Transaction
```

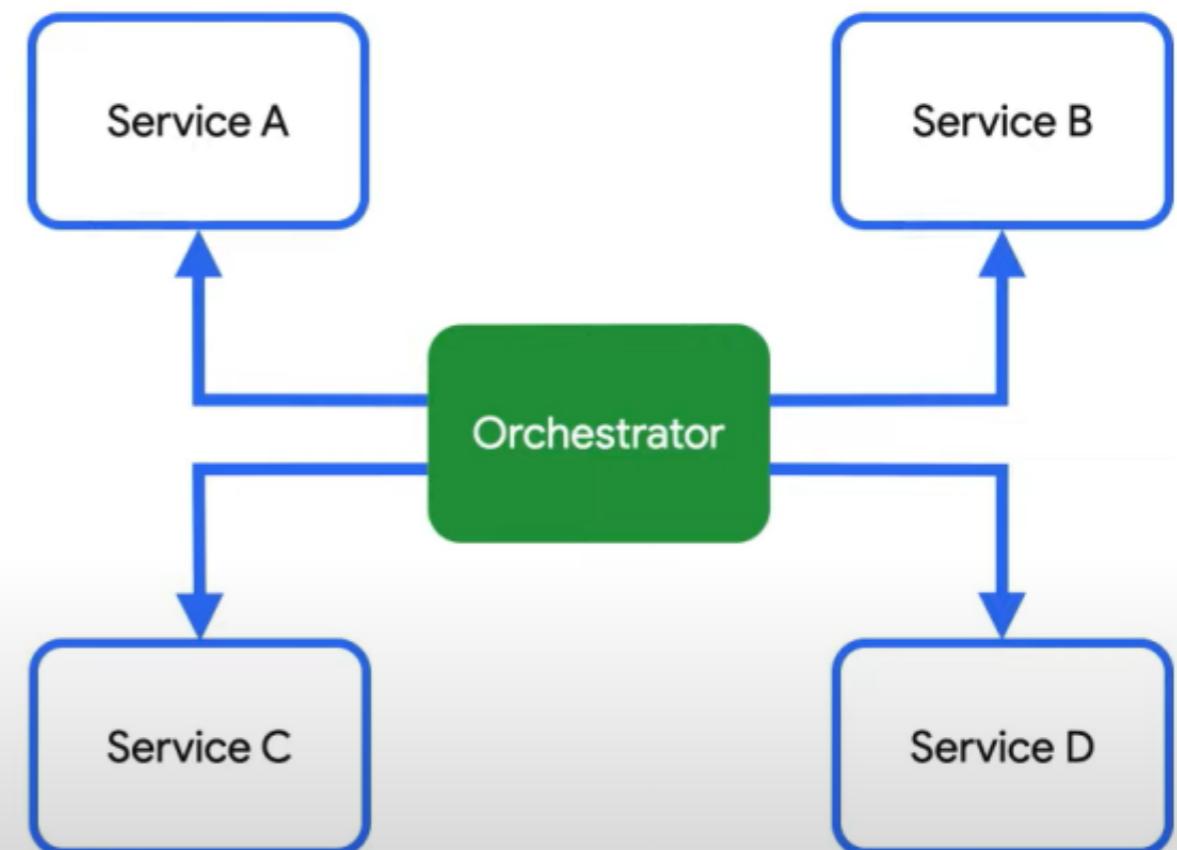


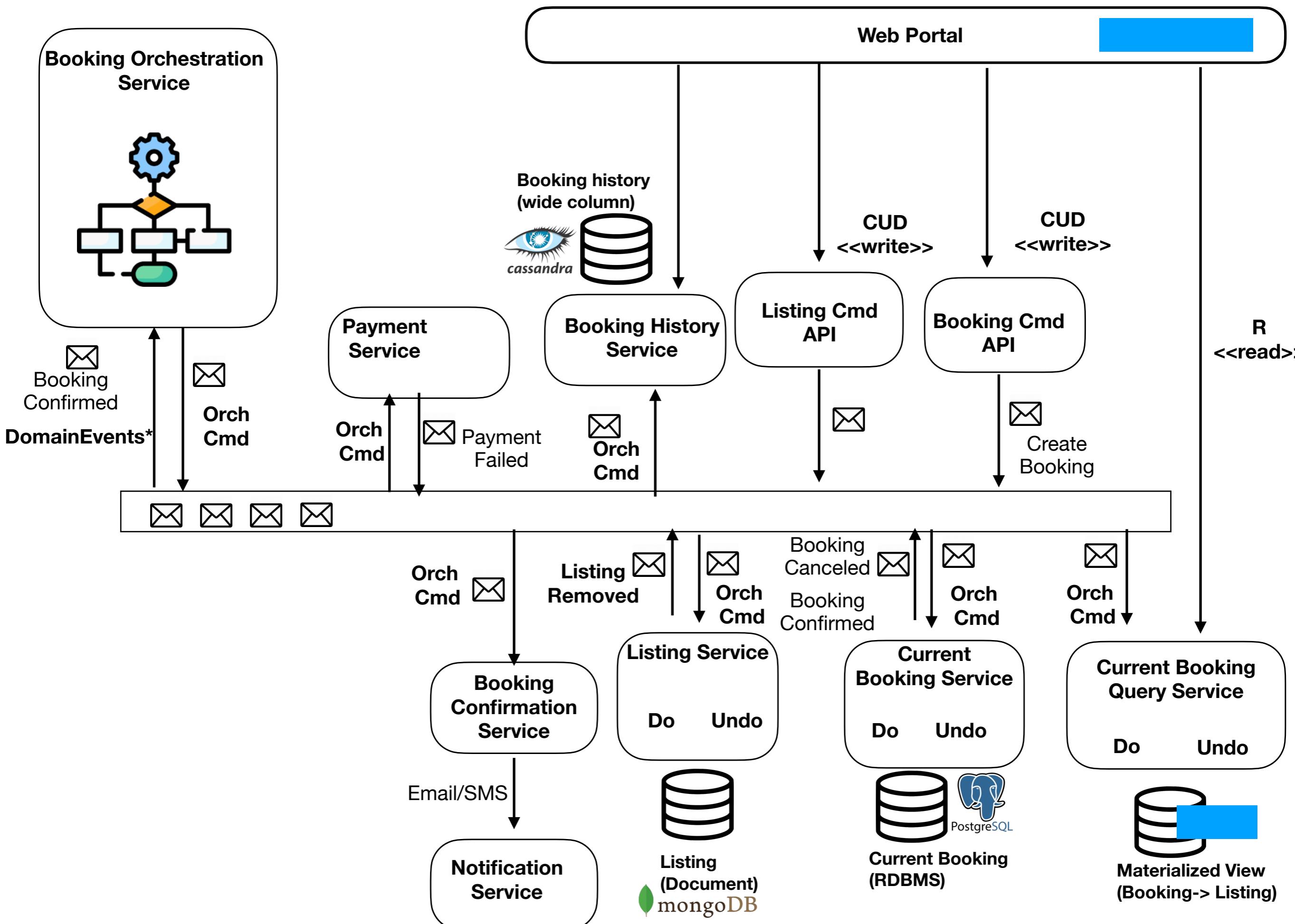


Choreography



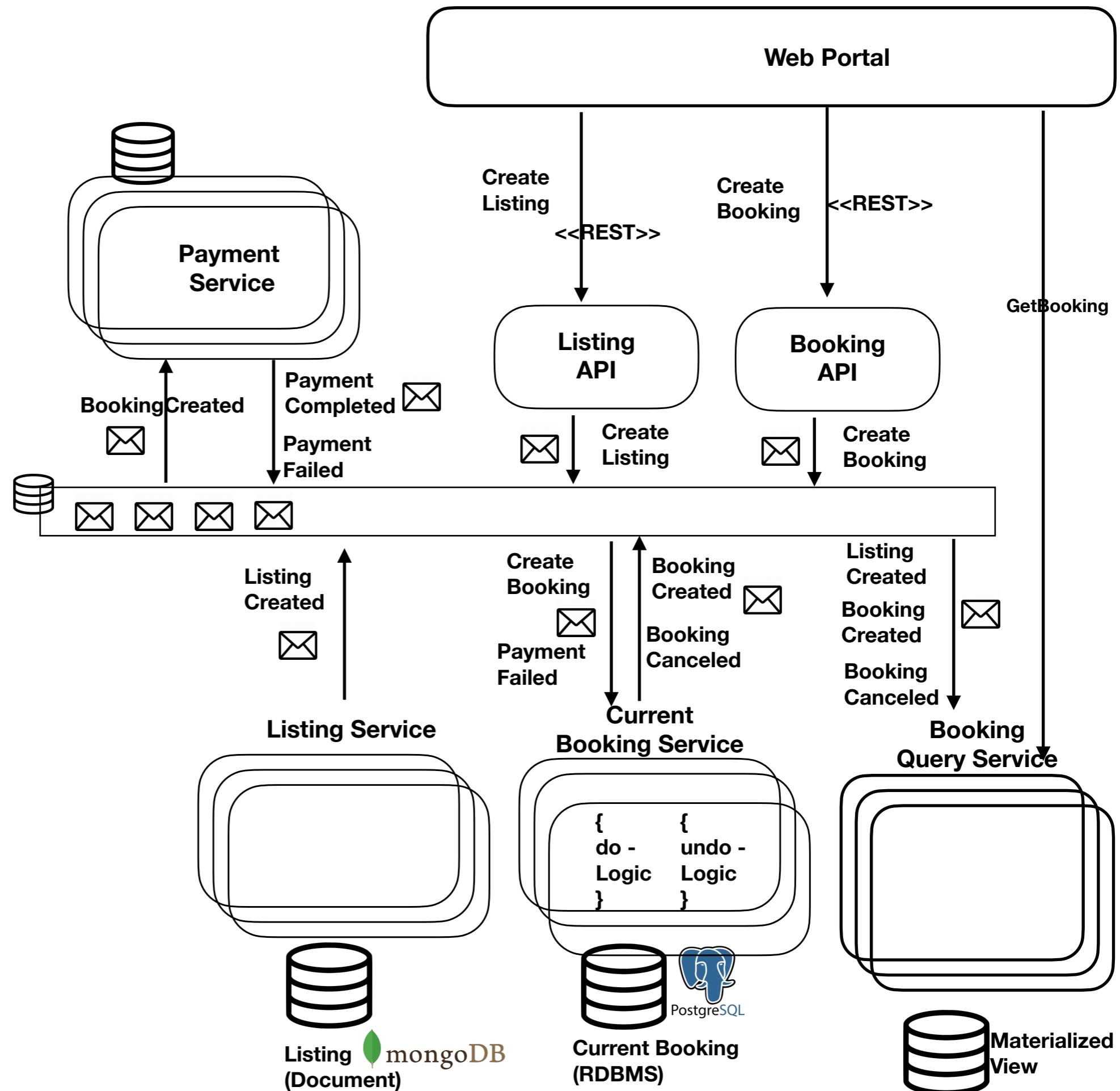
Orchestration



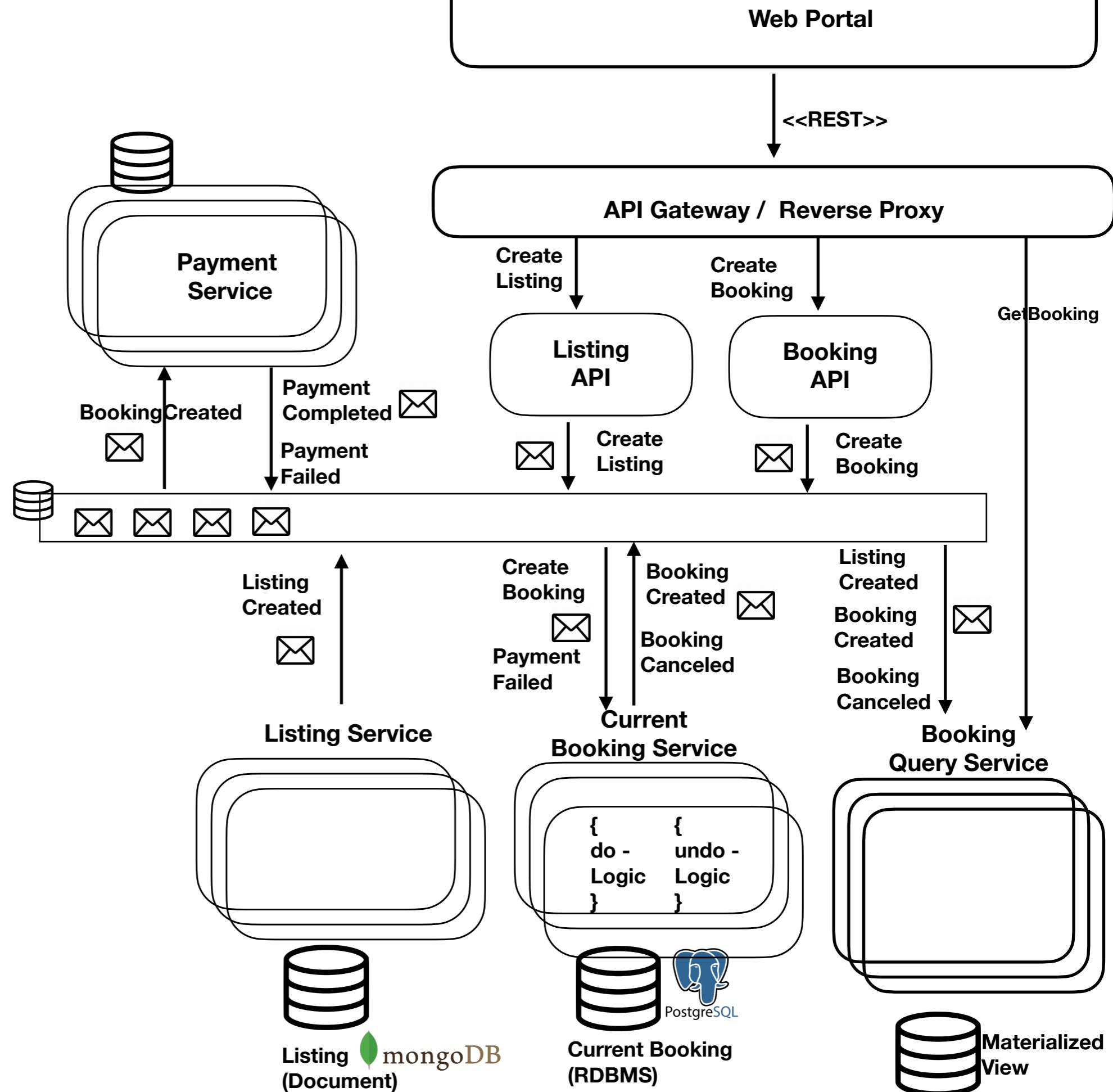


Gateway

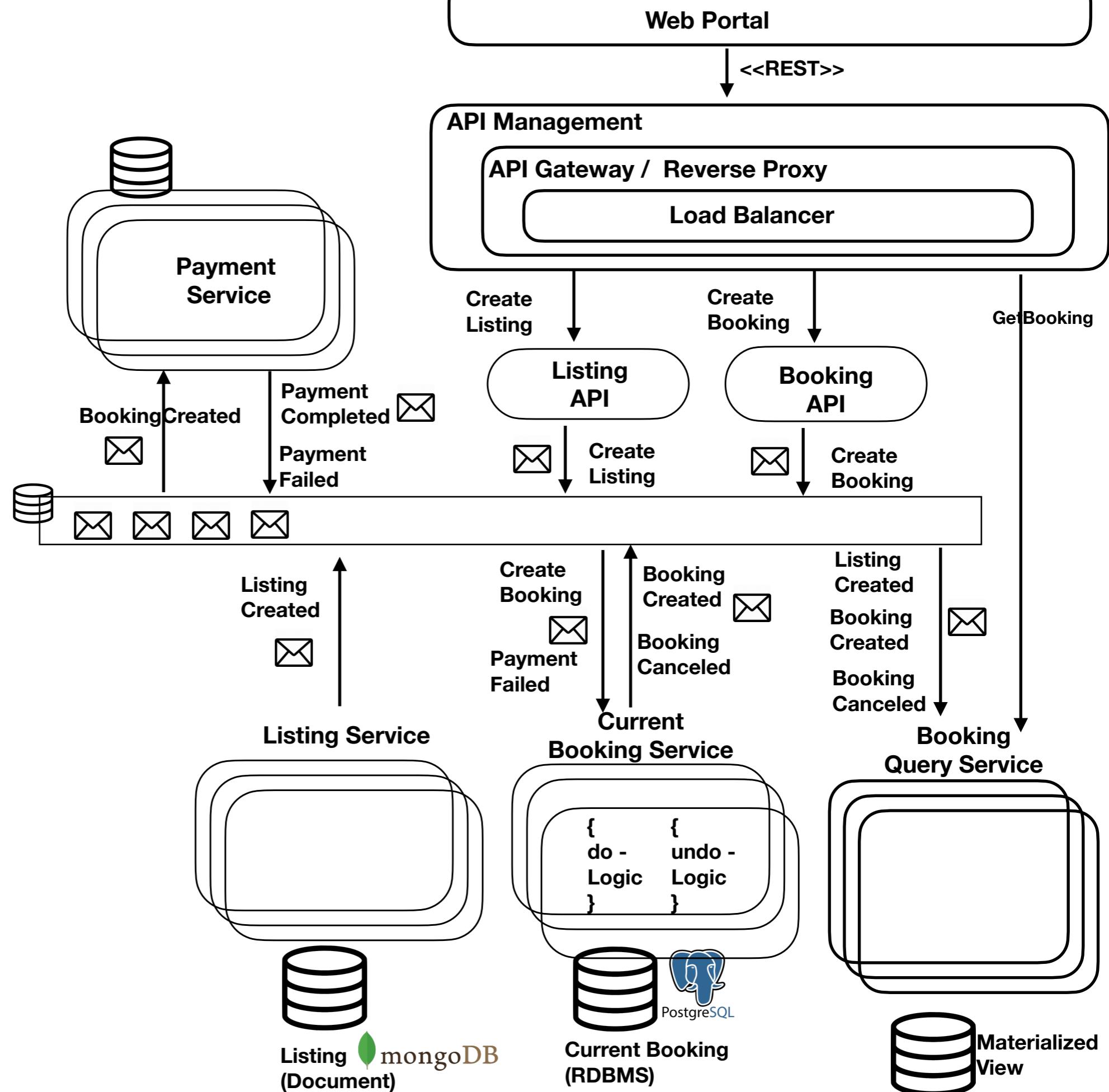
```
# Domain Command
# Domain Events
# Domain Exception
# Compensatable Transaction
```



API Gateway
reverse proxy
API Management

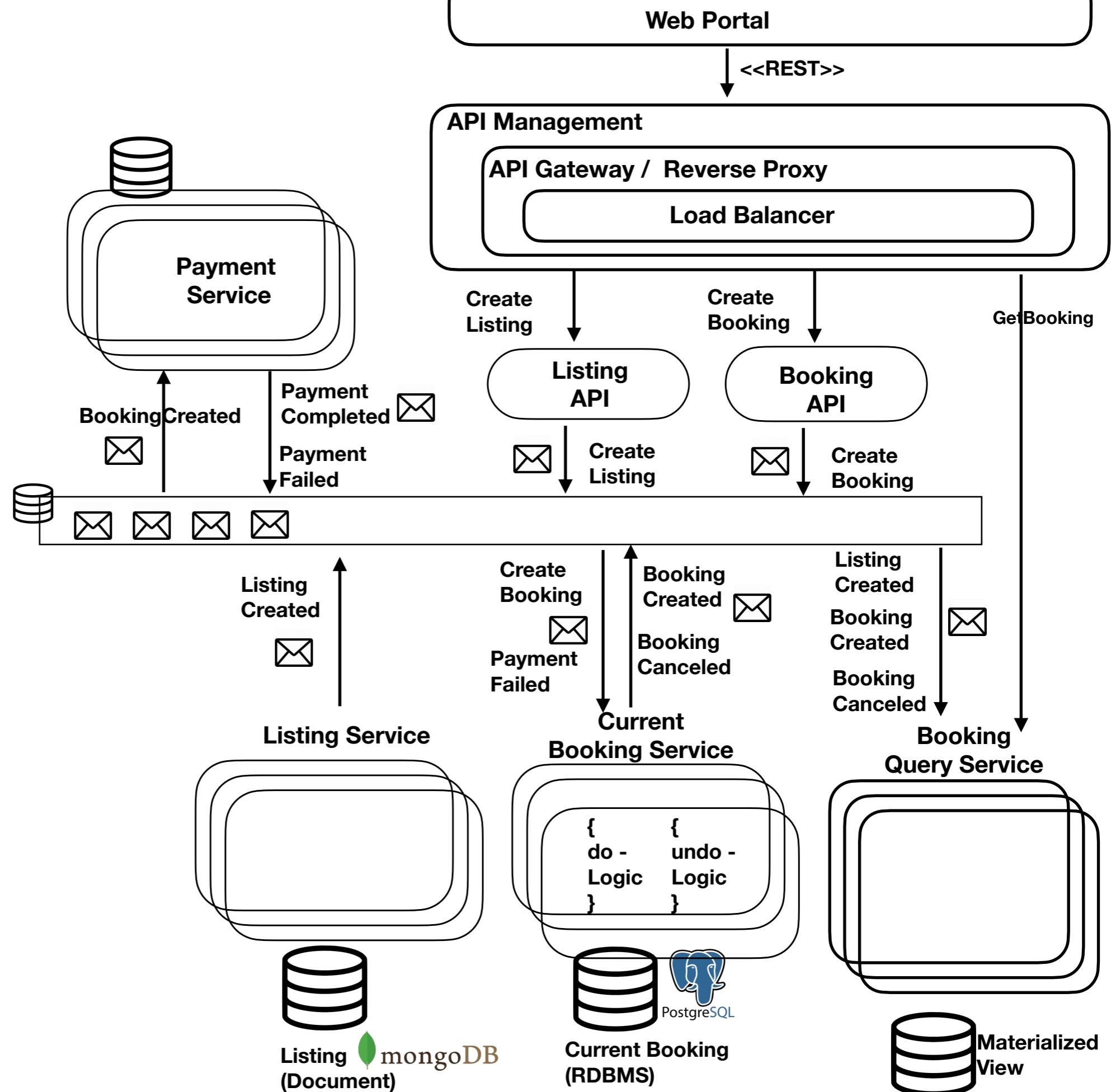


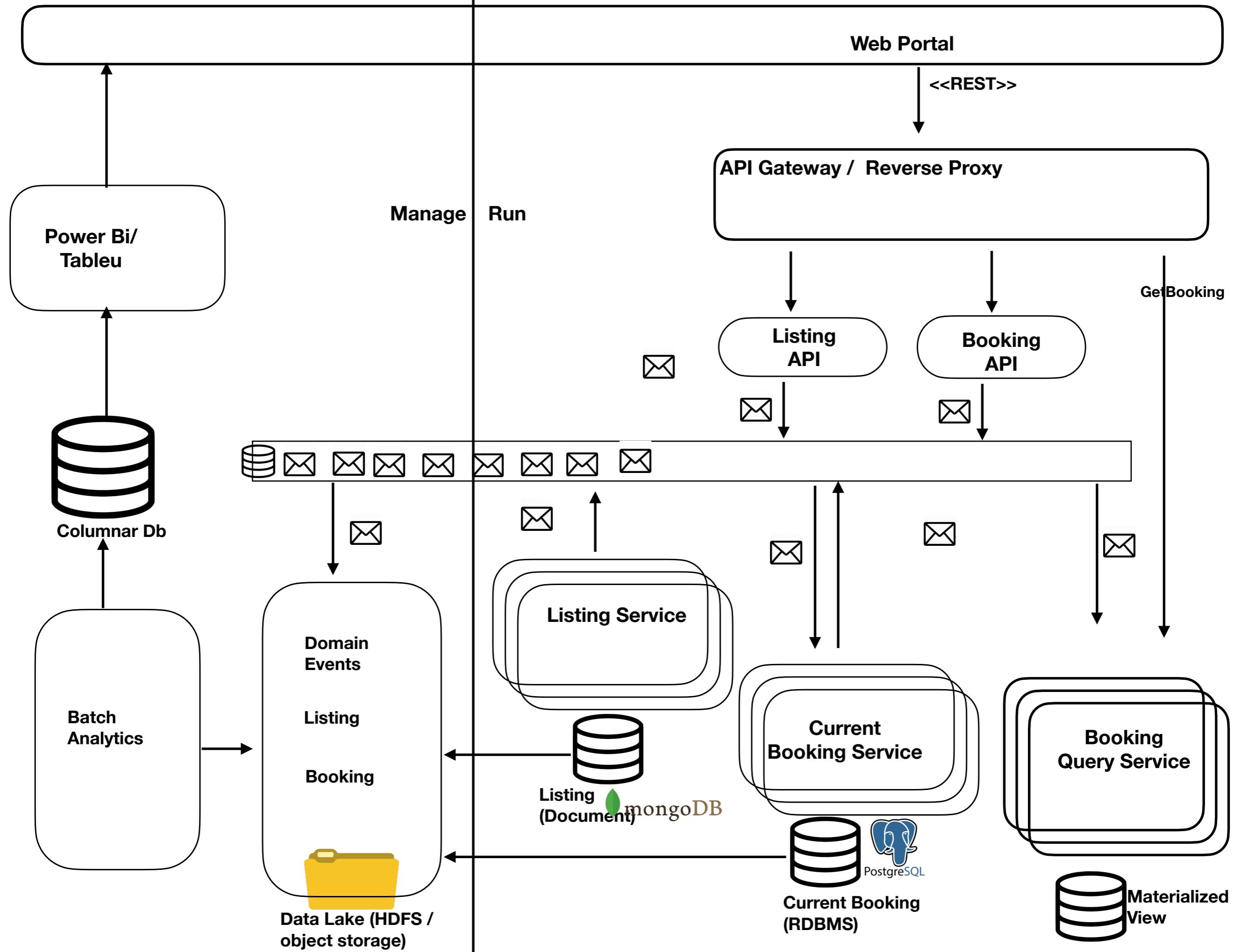
API Gateway
reverse proxy
API Management

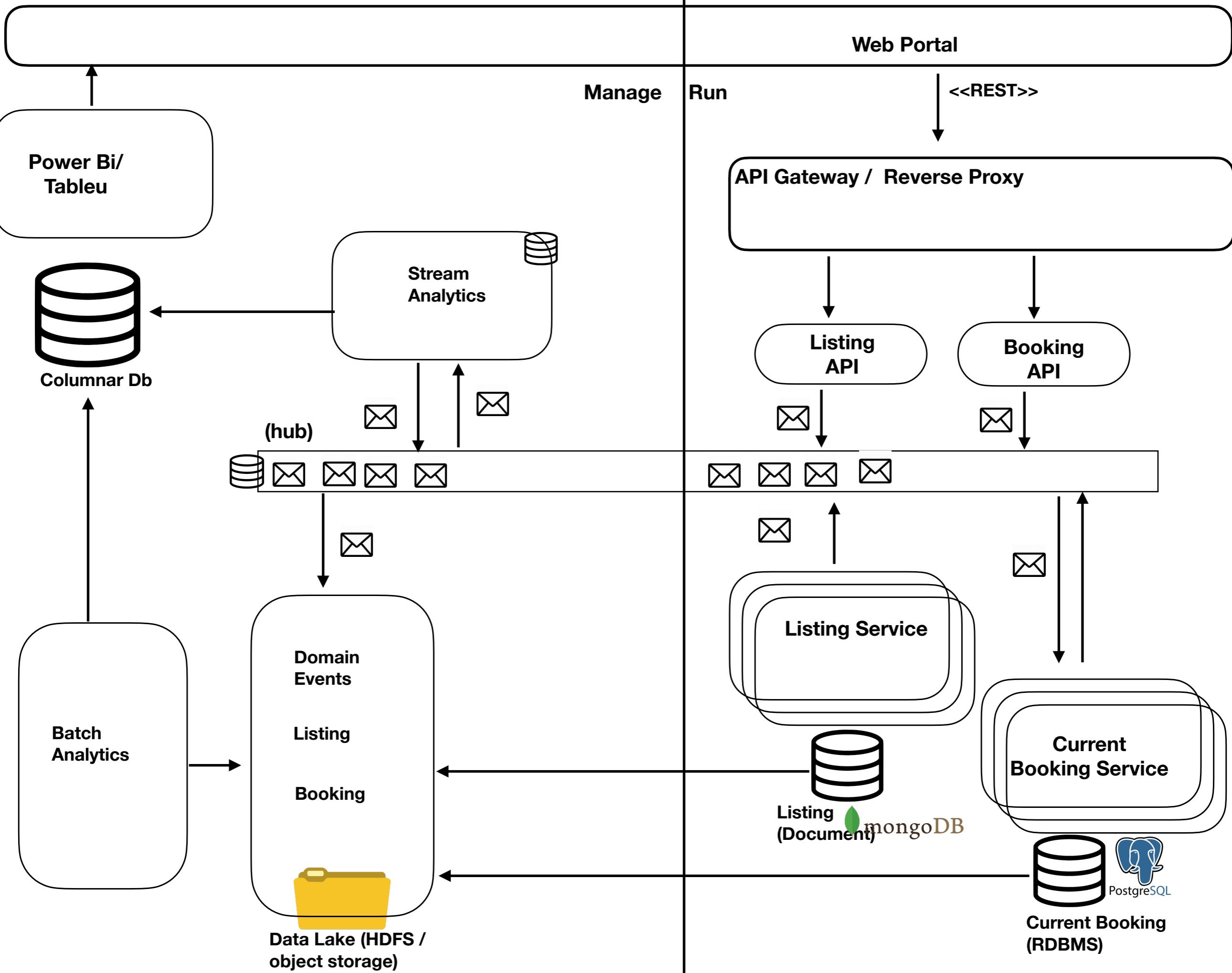


Analytics

API Gateway
reverse proxy
API Management

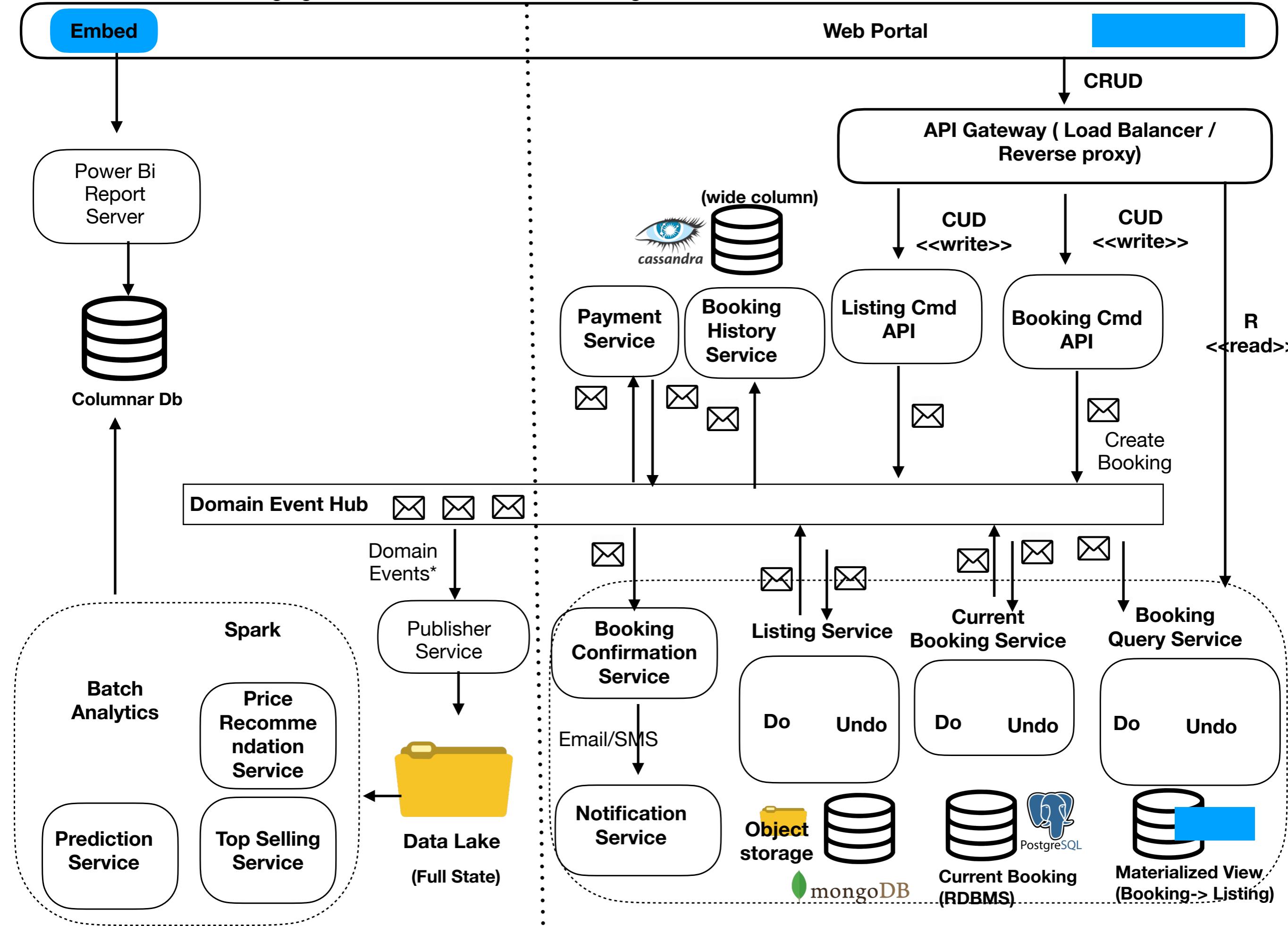






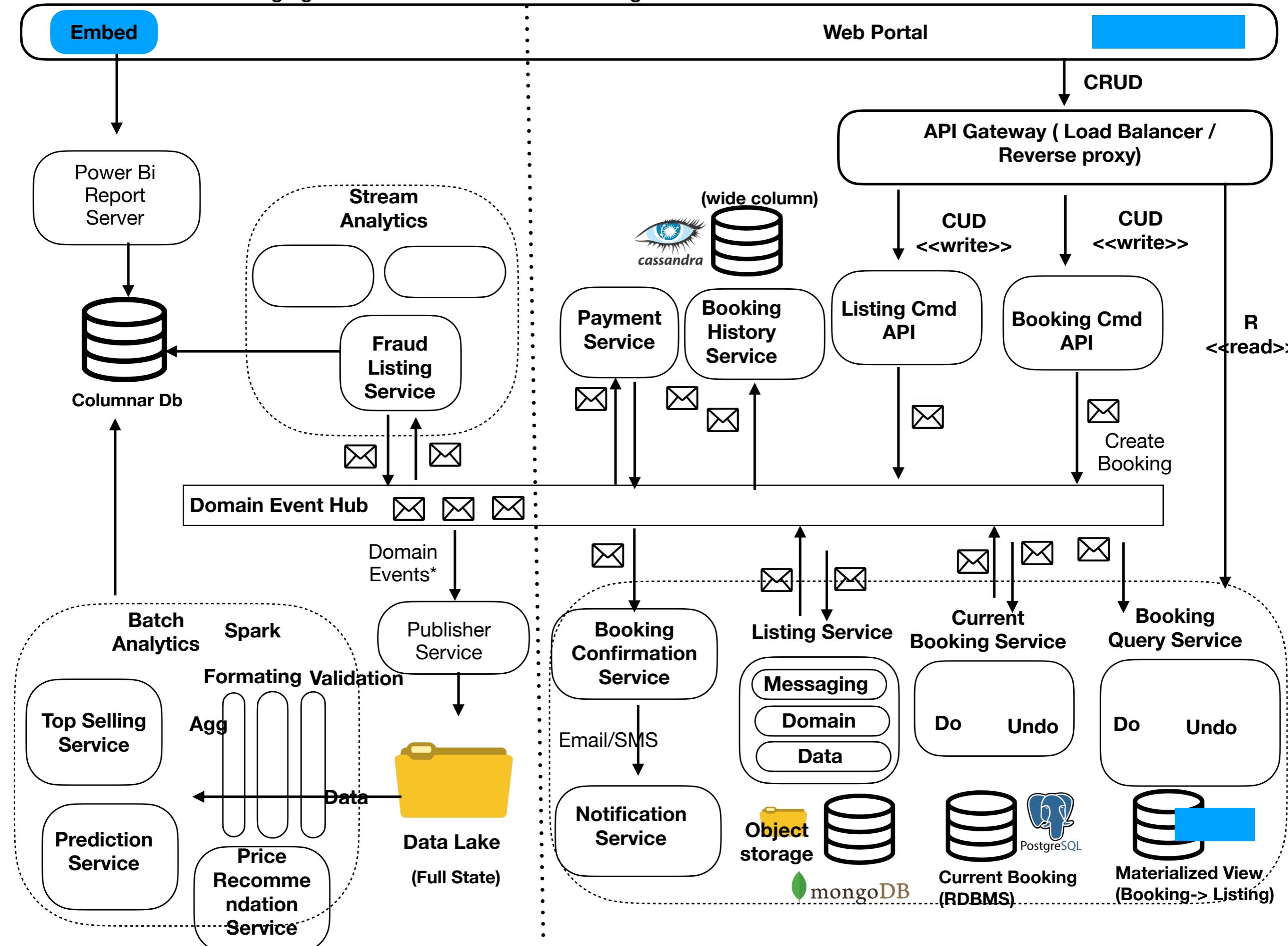
Managing Domain Architecture

Running Domain Architecture

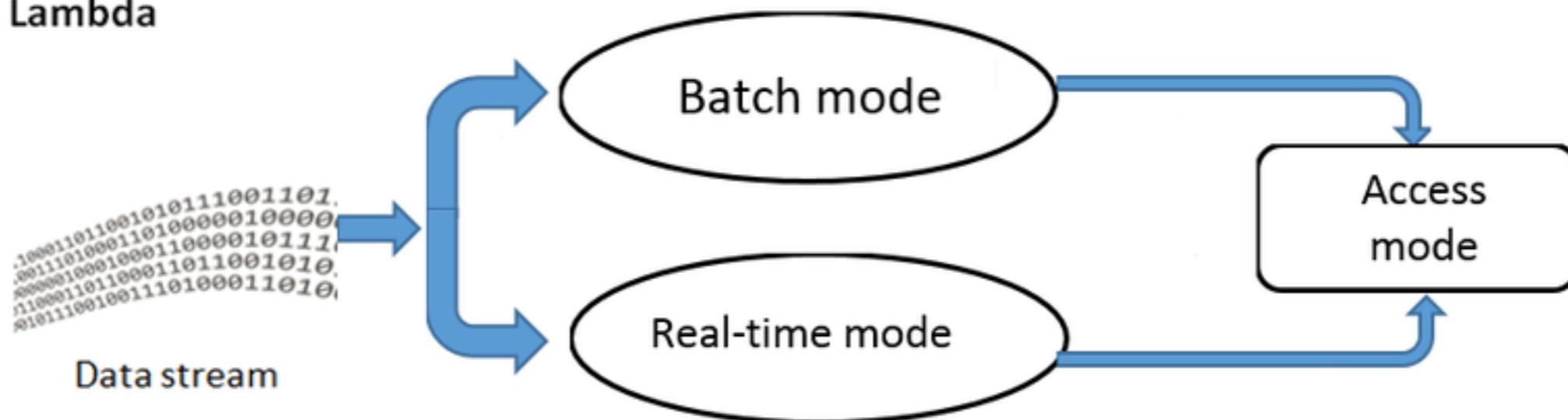


Managing Domain Architecture

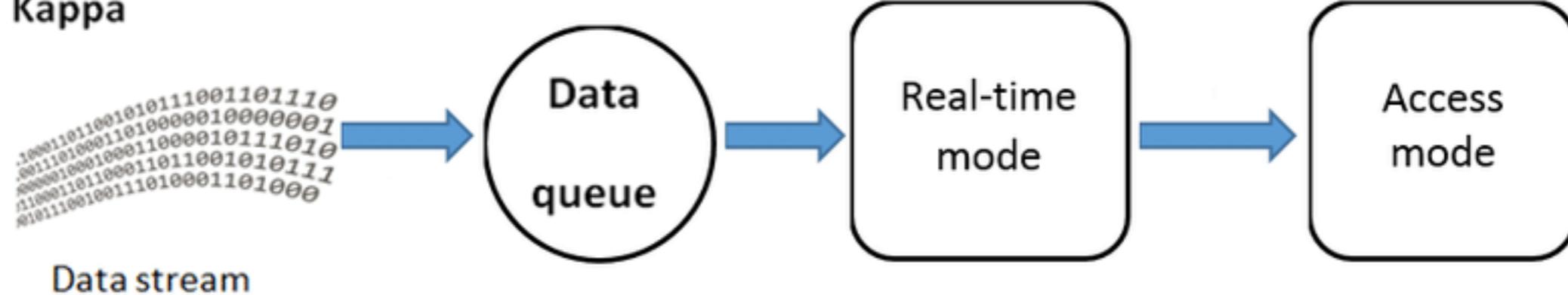
Running Domain Architecture



Lambda

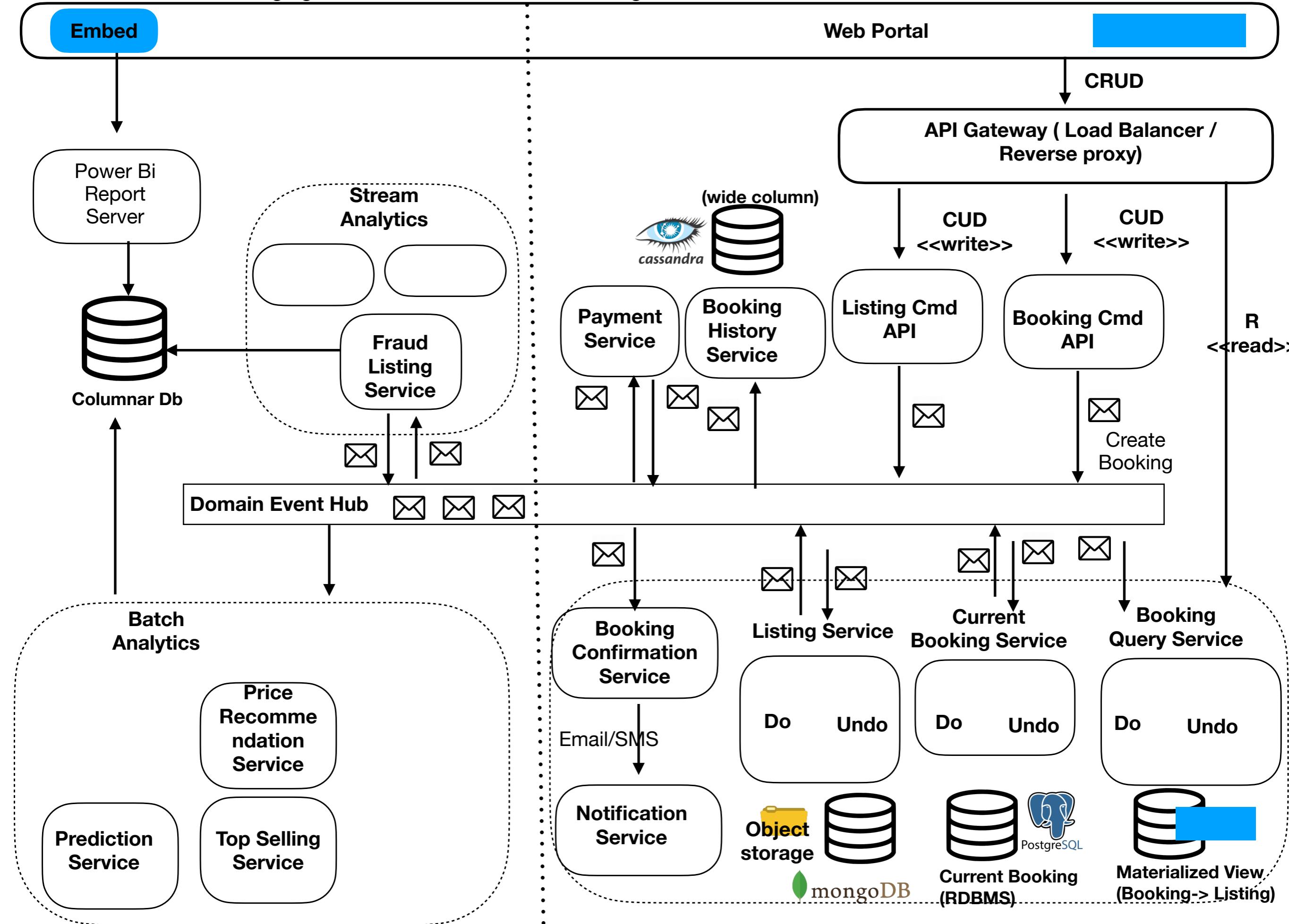


Kappa



Managing Domain Architecture

Running Domain Architecture



Managing Domain Architecture

Running Domain Architecture

Embed

Power Bi
Report
Server

Stream
Analytics

Lake house
Databricks
dreamio

Event Hub

Domain
Events*

Batch
Analytics

Spark

Price
Recommen
dation
Service

Top Selling
Service



Data Lake
(Full State)

Web Portal

CRUD

API Gateway (Load Balancer /
Reverse proxy)



Payment
Service

Booking
History
Service

Listing Cmd
API

Booking Cmd
API



Create
Booking

R
<<read>>



Booking
Confirmation
Service

Current
Booking
Service

Booking
Query Service

Email/SMS

Notification
Service

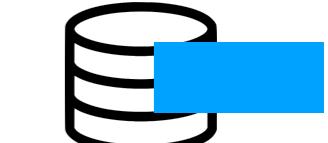
Do Undo

Do Undo

Do Undo



Current Booking
(RDBMS)



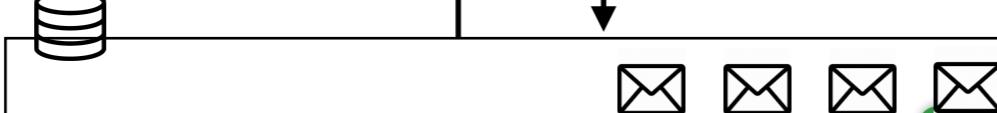
Big Data

Manage Business

Run Business

Reporting
Power bi/ Tableau/
SuperSet/ ...

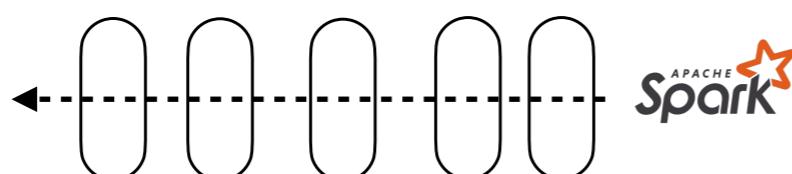
Stream
Analytics



Domain
Events*

Event
Service

BatchAnalytics



Gold Zone



Columnar
(Star schema)

Staging Zone



(HDFS / S3 / Blob)

Landing Zone



(HDFS / S3 / Blob)

Data Lake

Csv, json, parquet, ice berg

Web Portal

Domain
Services



Transaction
Db

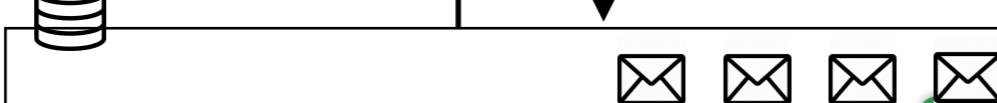
Ingress

Manage Business

Run Business

Reporting
Power bi/ Tableau/
SuperSet/ ...

Stream
Analytics



Domain
Events*

Event
Service

BatchAnalytics
(Databircks)



Gold Zone



Columnar
(warehouse)
(Star schema)

Staging Zone



(HDFS / S3 / Blob)

Landing Zone

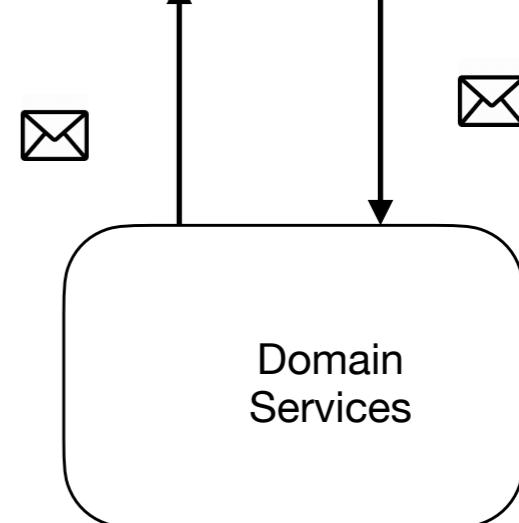


(HDFS / S3 / Blob)

Data Lake

Csv, json, parquet, ice berg

Web Portal



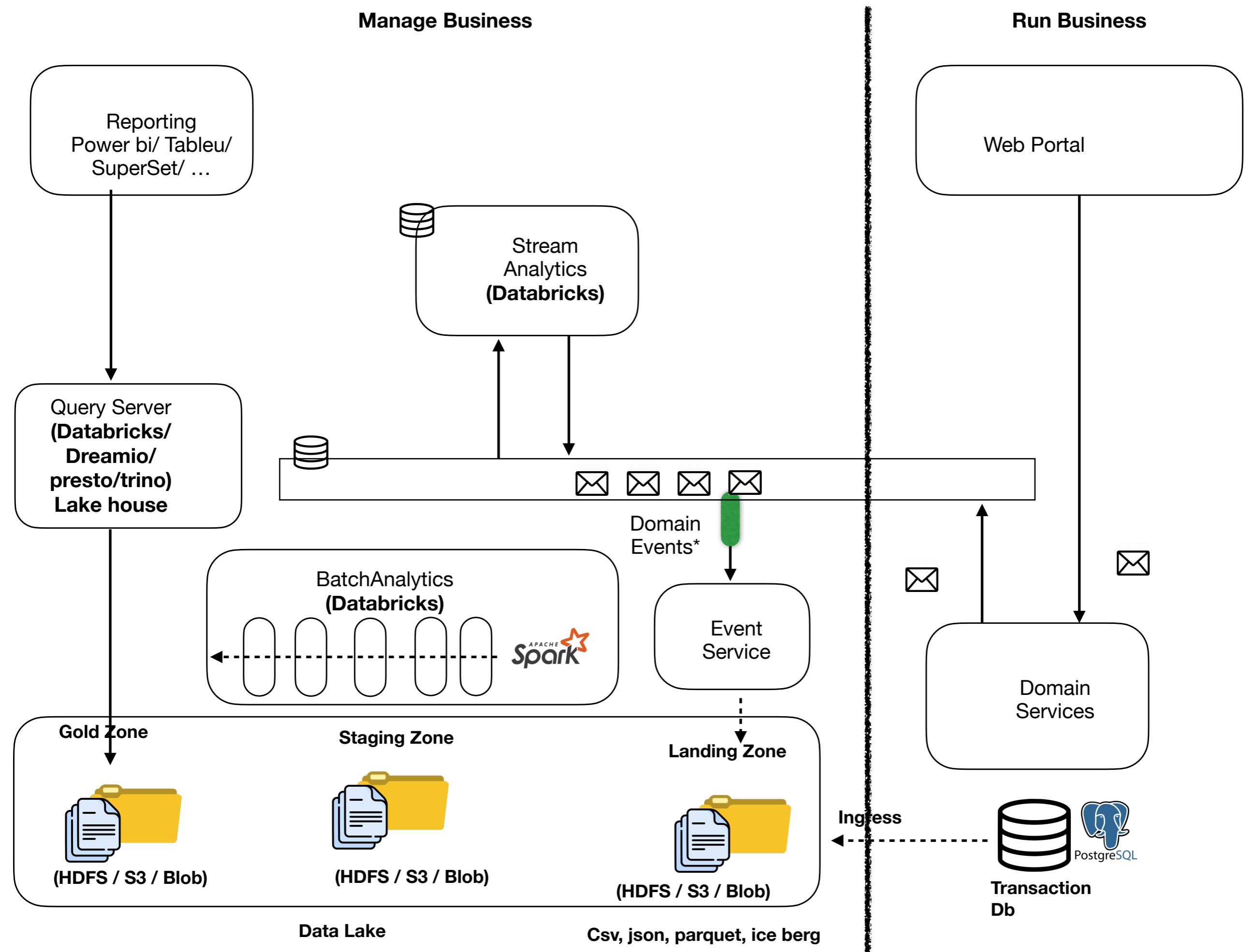
Ingress

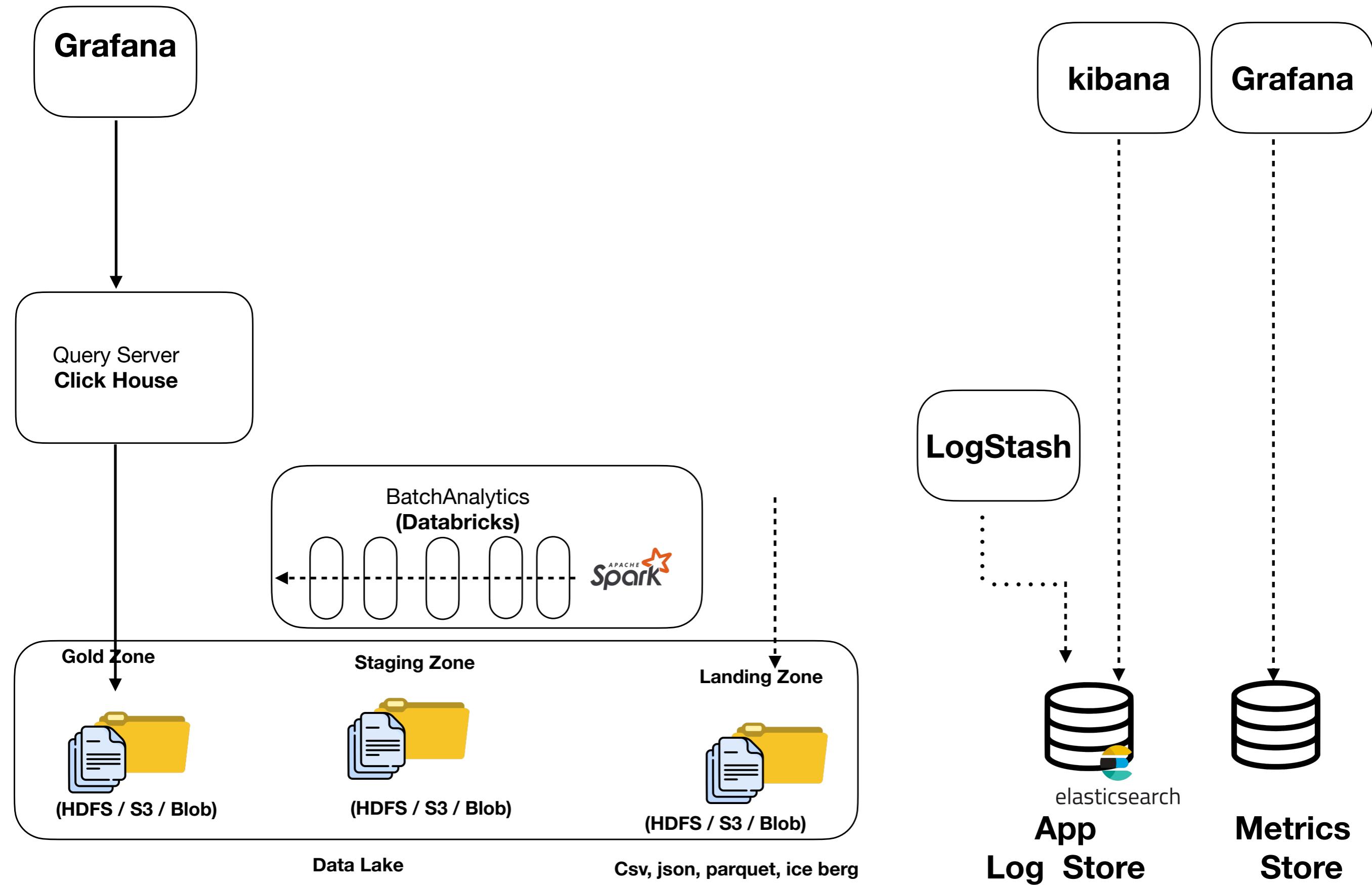


Transaction
Db

Manage Business

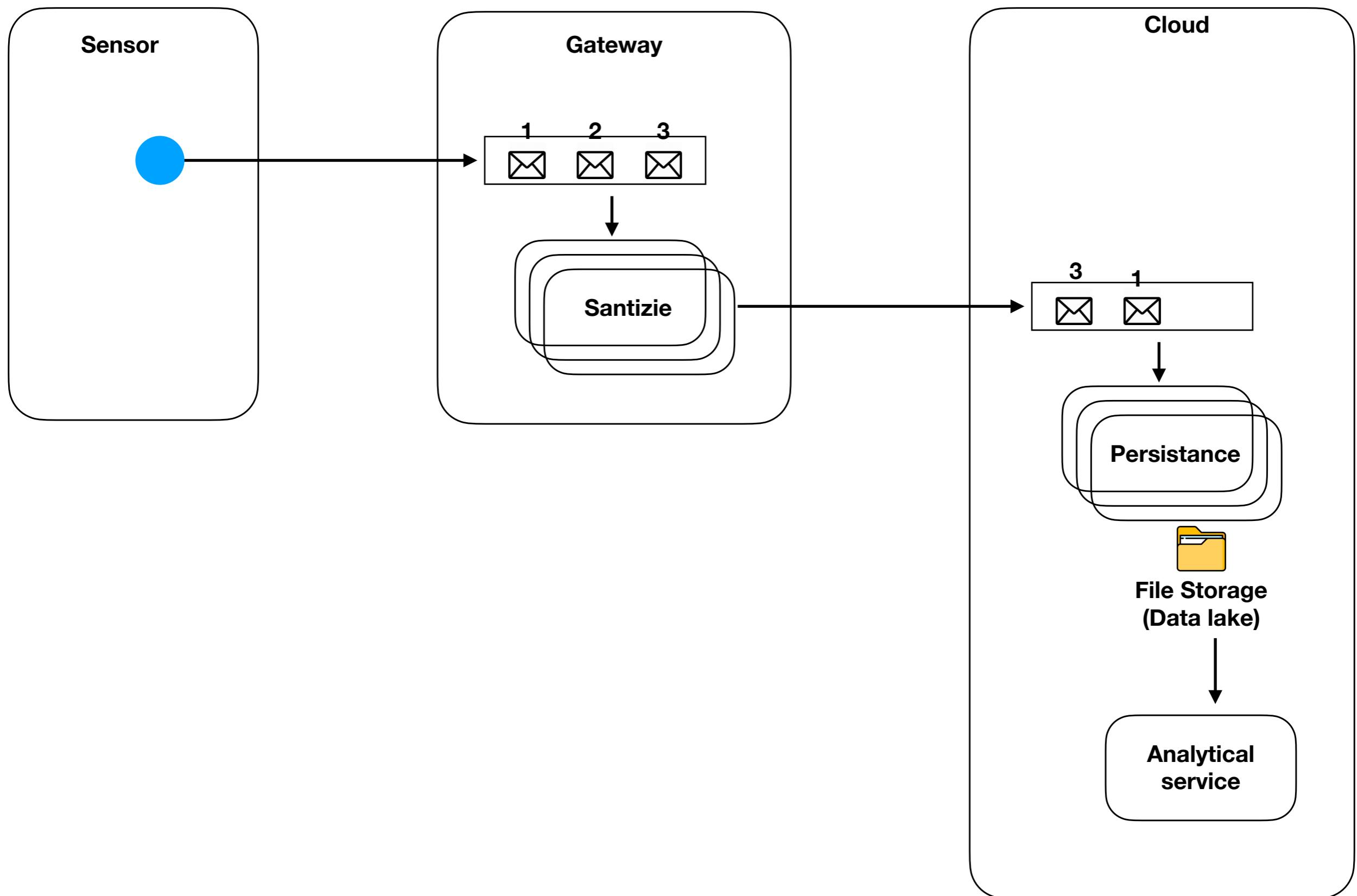
Run Business





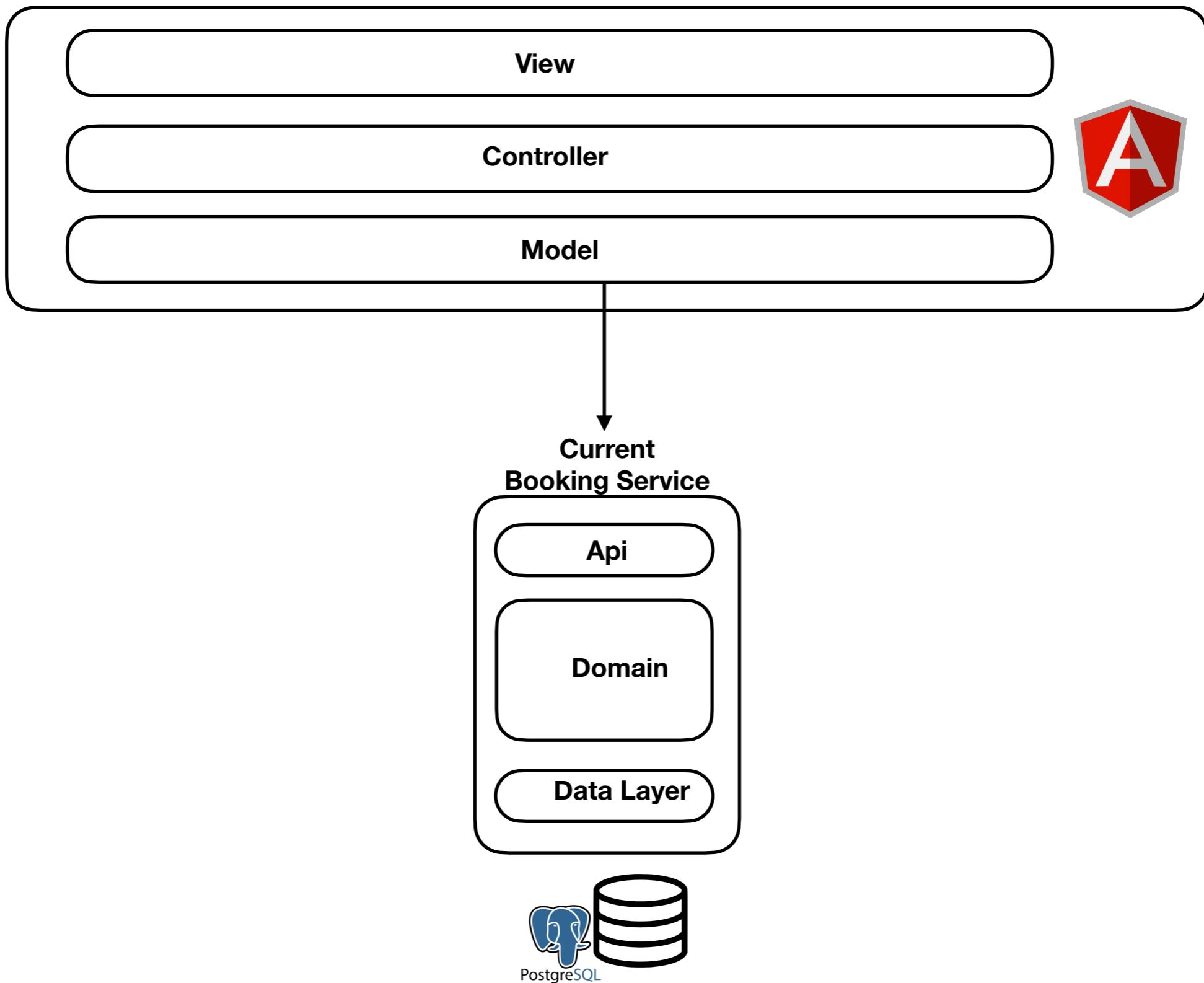
View - Unordered

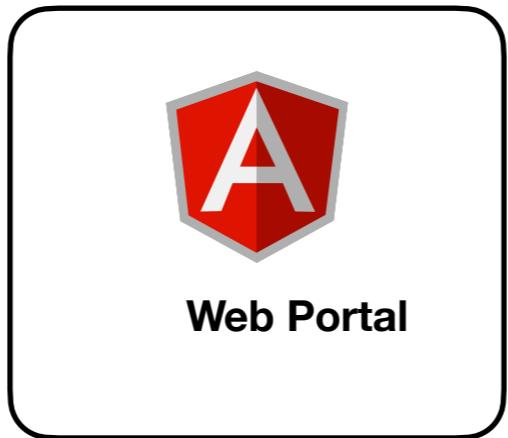
IOT



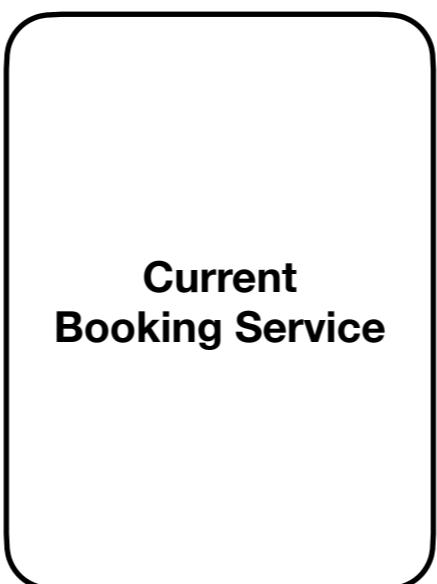
View - Queue

Web Portal



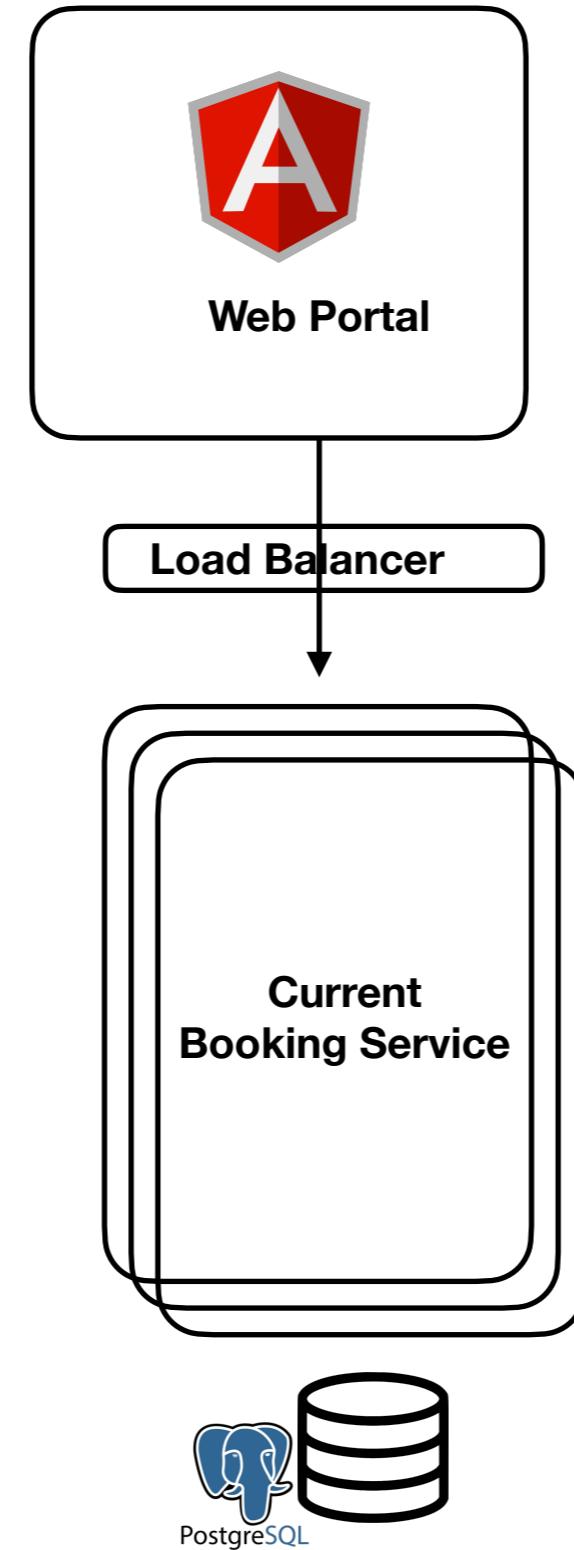


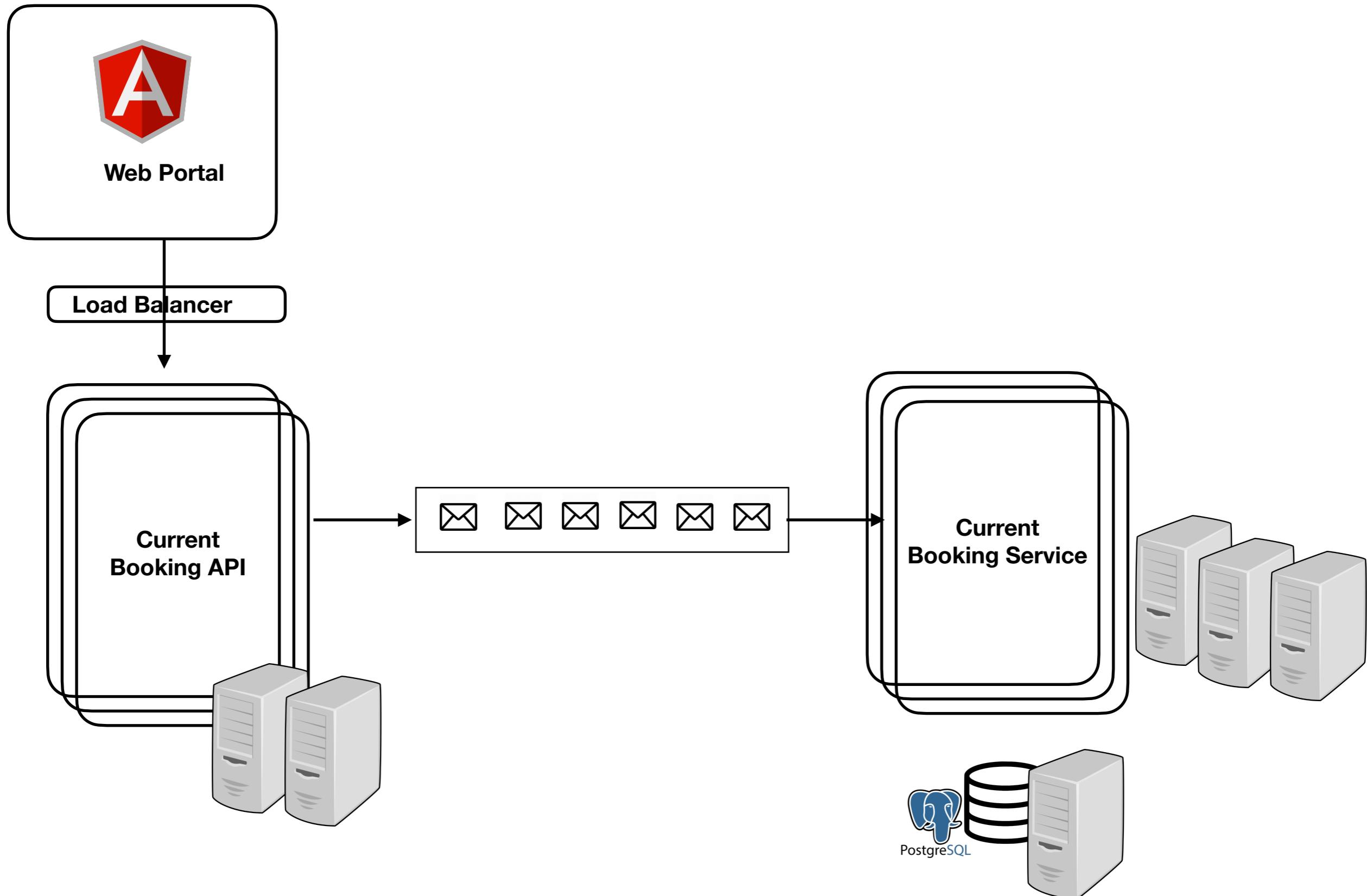
Web Portal



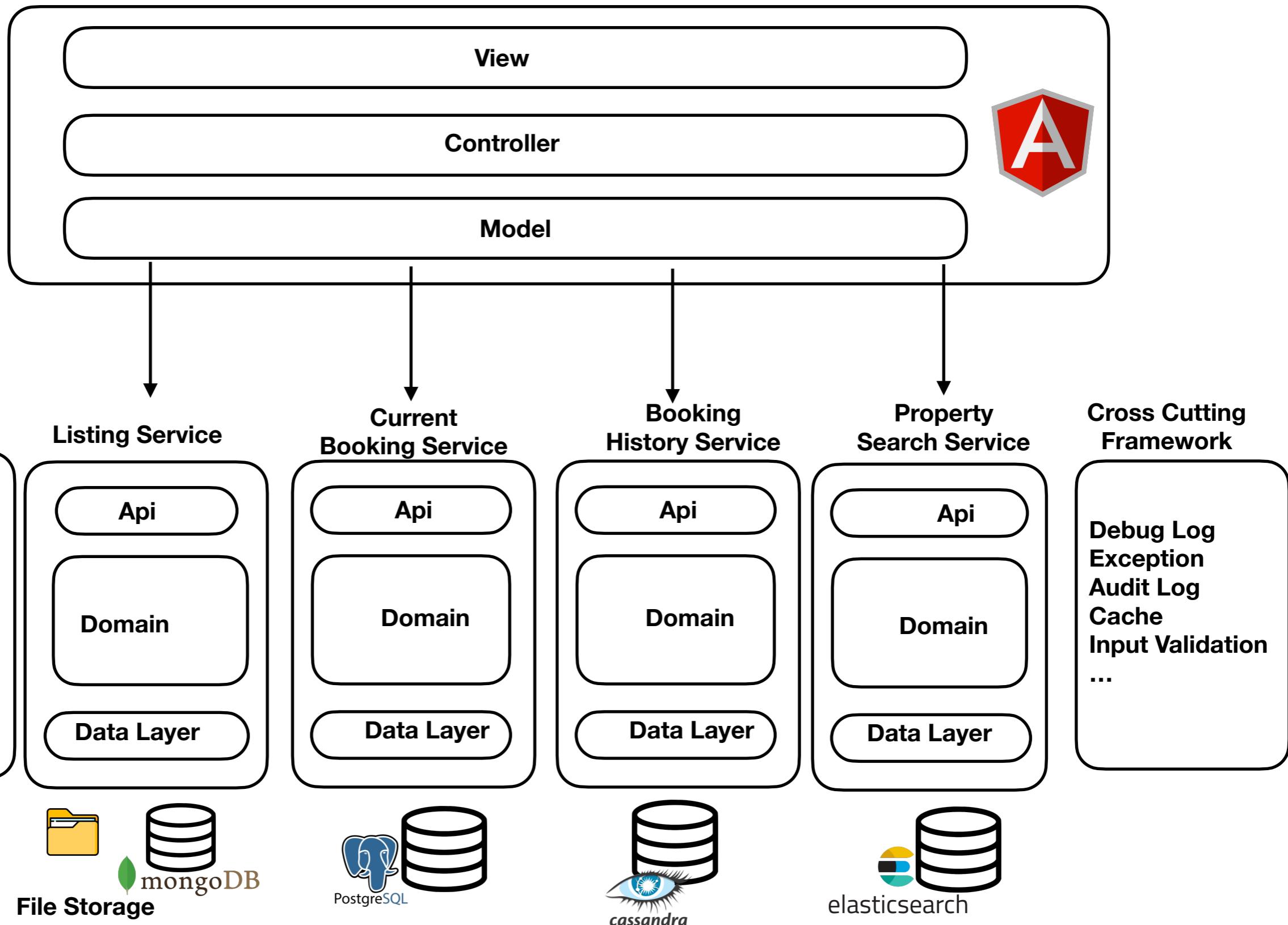
**Current
Booking Service**







Web Portal



API

*** completes in next few seconds**

Immediate

Pessimistic

**# redbus
book my show**

Optimistic

irctc

*** locks before
transaction**

Messaging

*** completes in next few minutes/hours**

Eventual

*** locks within
transaction**

Data

Application Type

Java, Native, Web, Mainframe, Electron,

Functionality Type

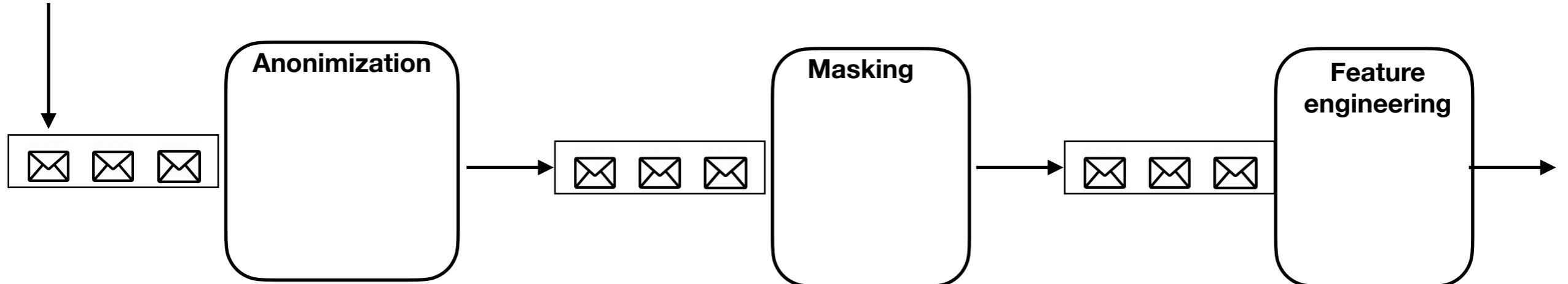
Anonimization, Masking, Feature engineering, ...

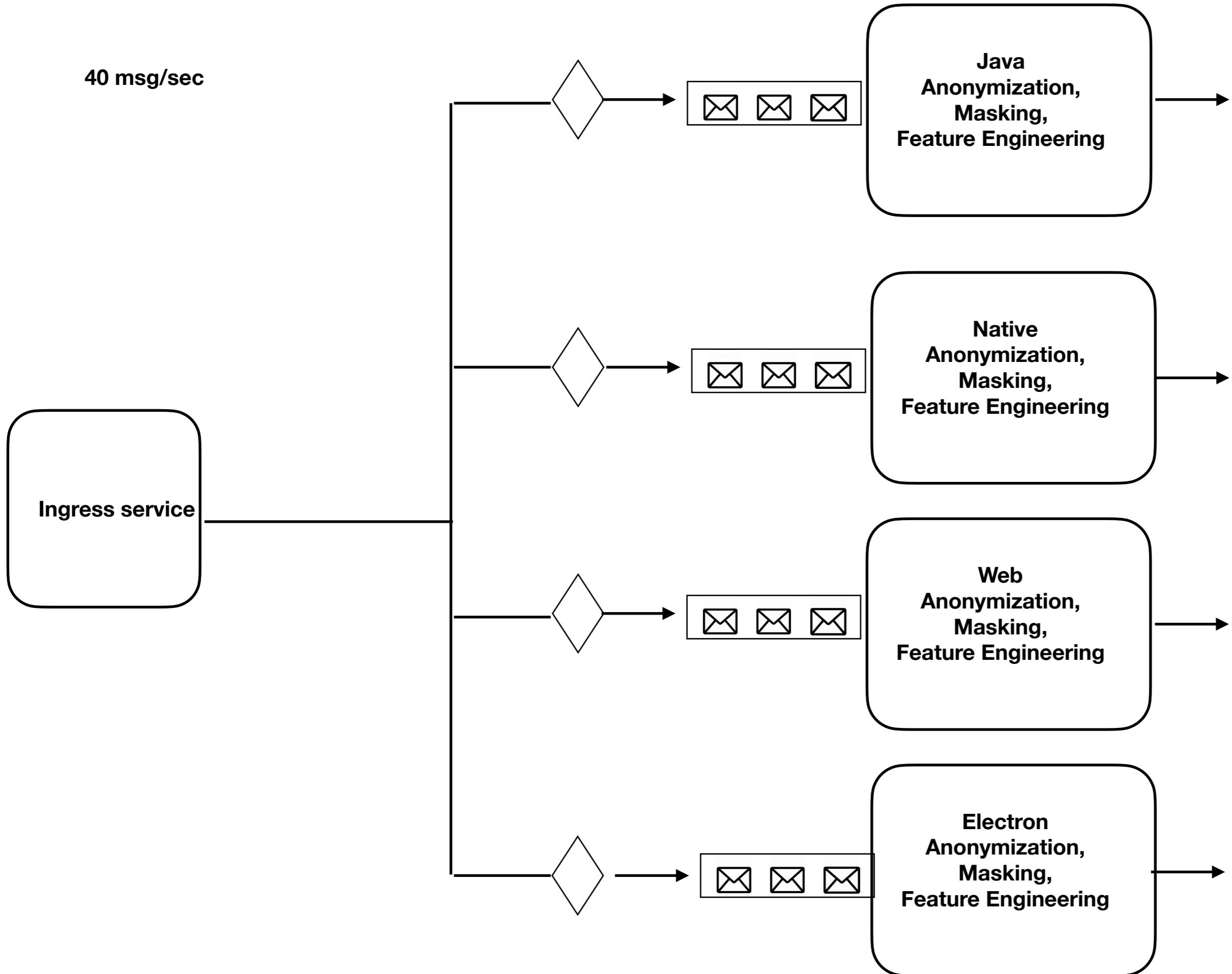
Java,
Anonymization

Java
Anonymization,
Masking,
Feature Engineering

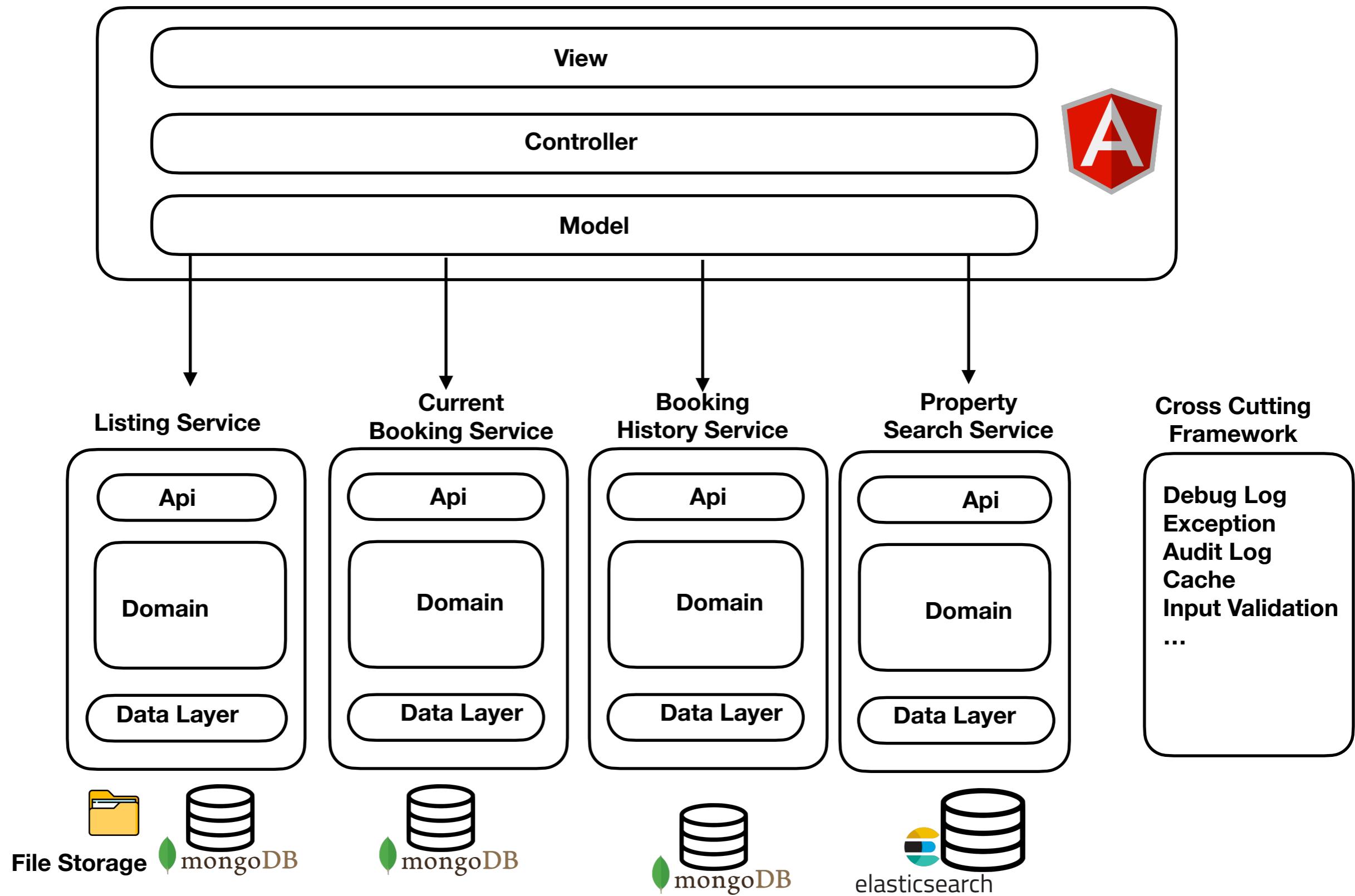
Anonymization
Java,
Native,
Web
Mainframe,
Electron

8 msg/sec

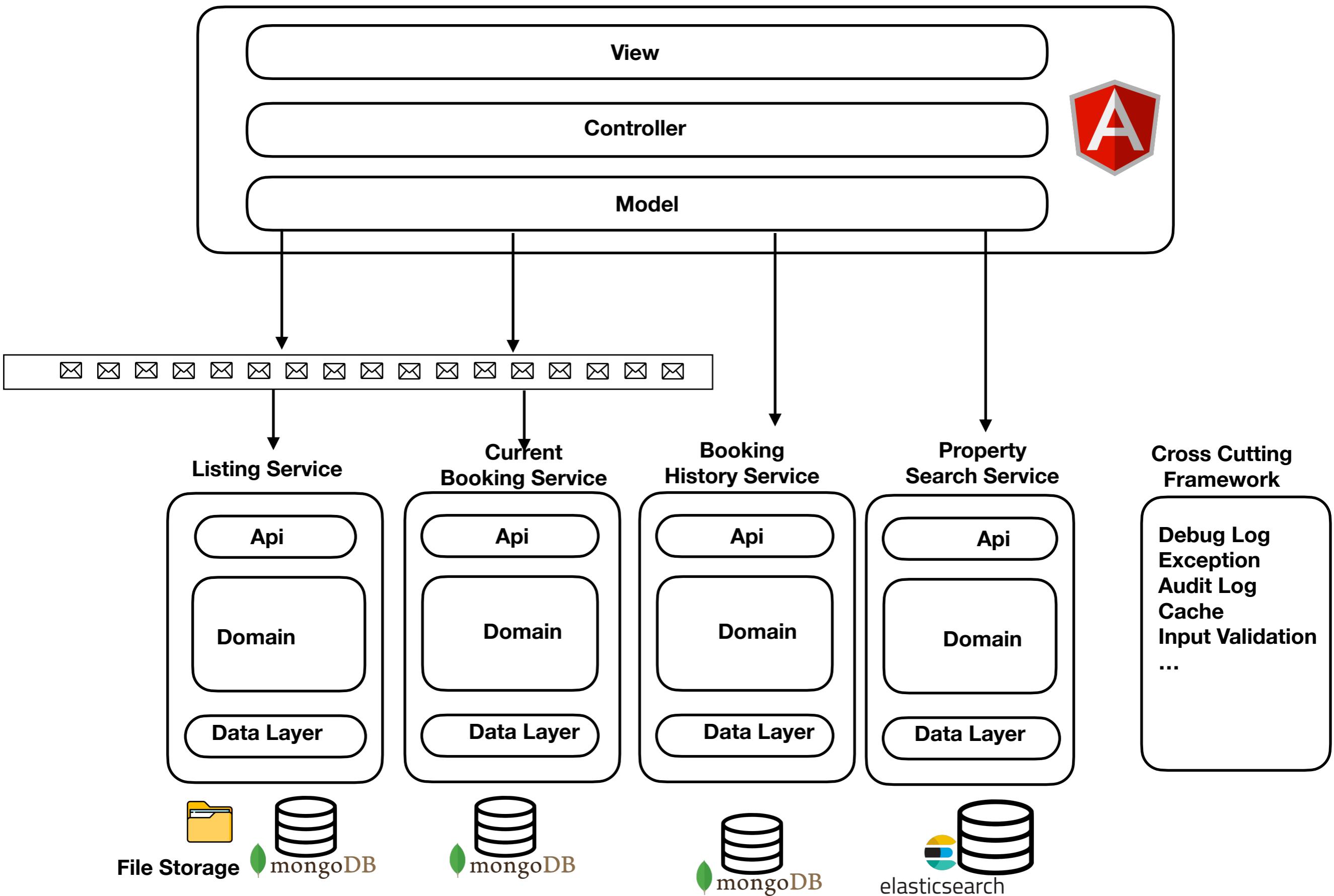


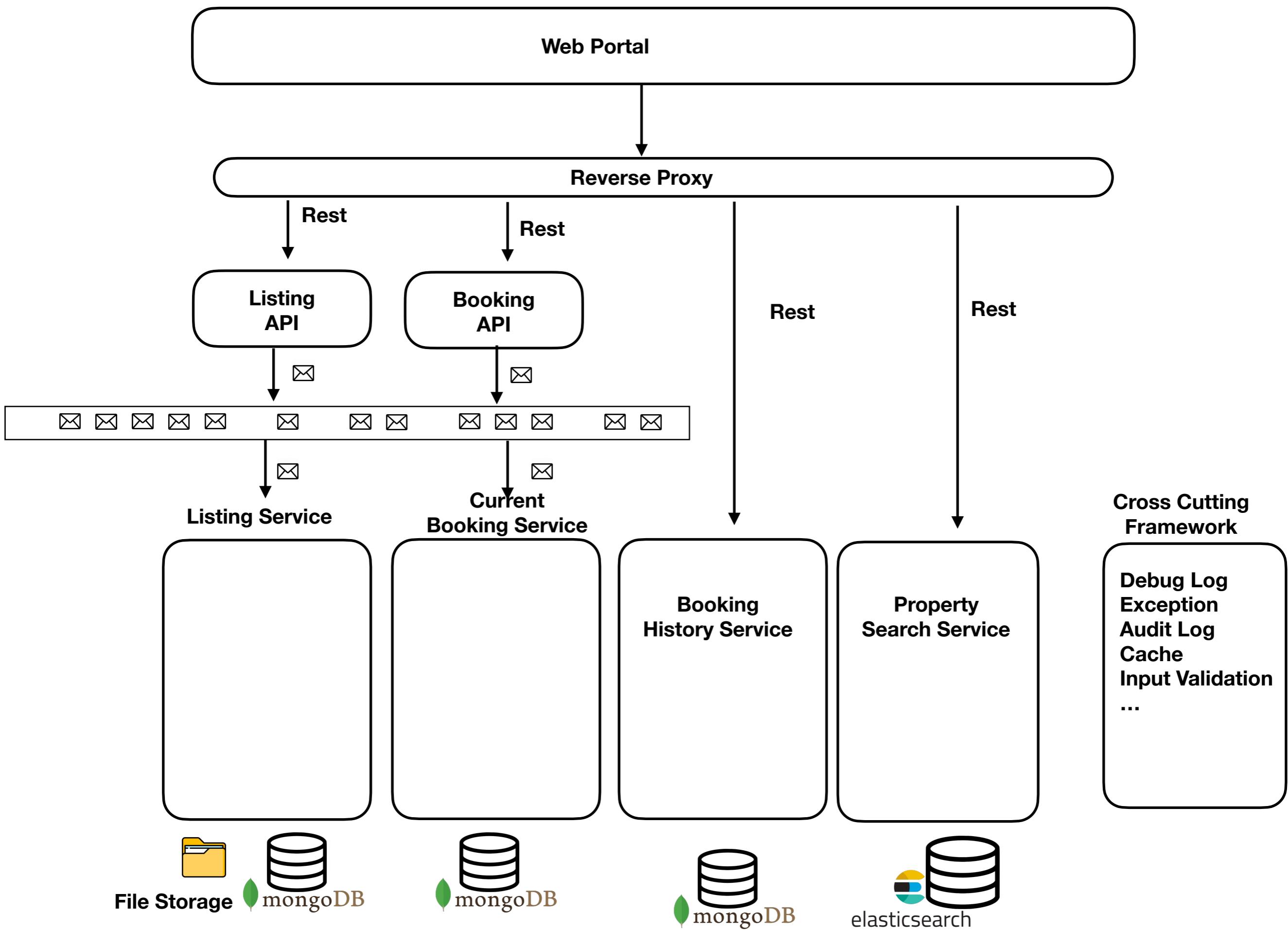


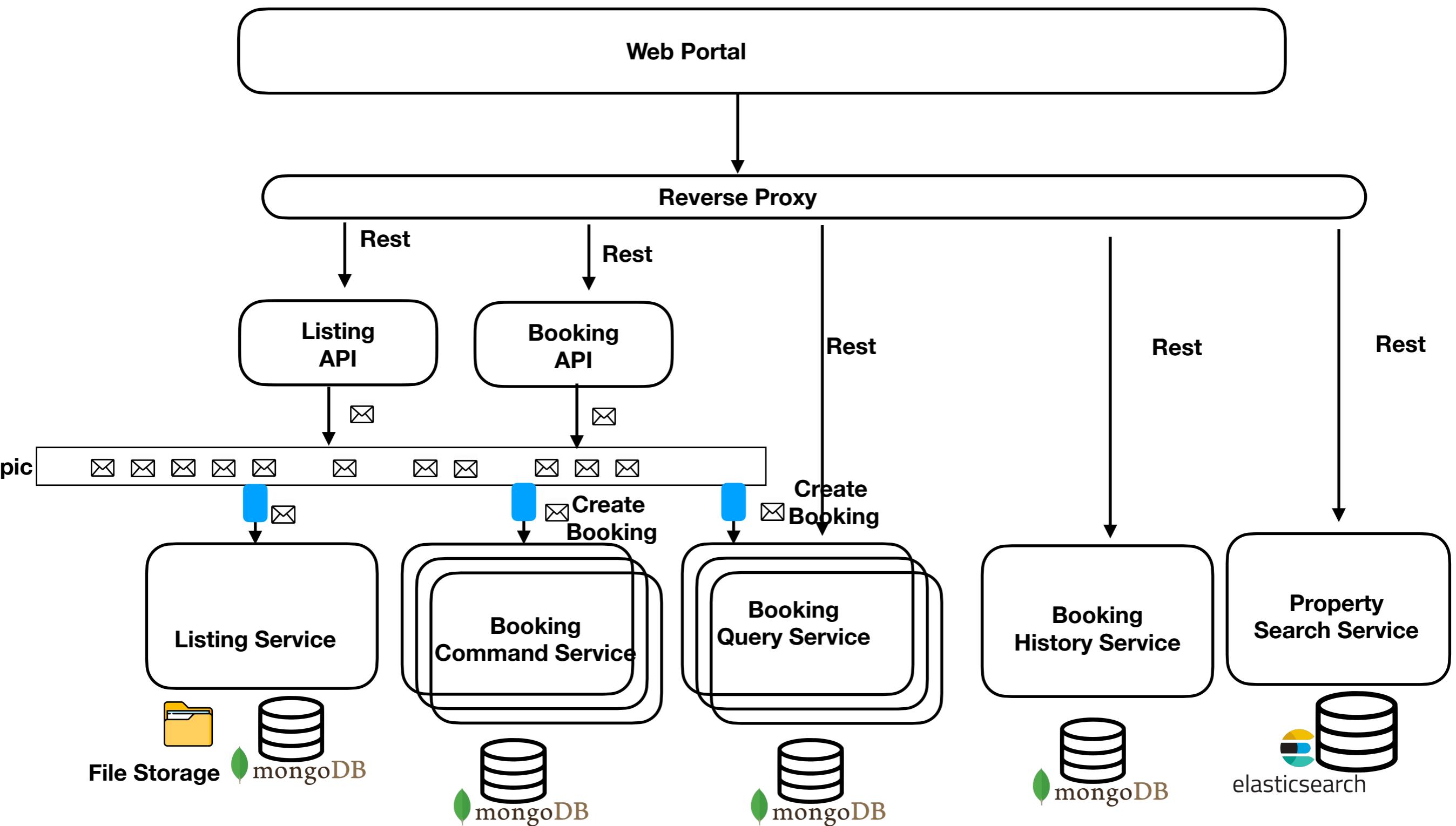
Web Portal



Web Portal

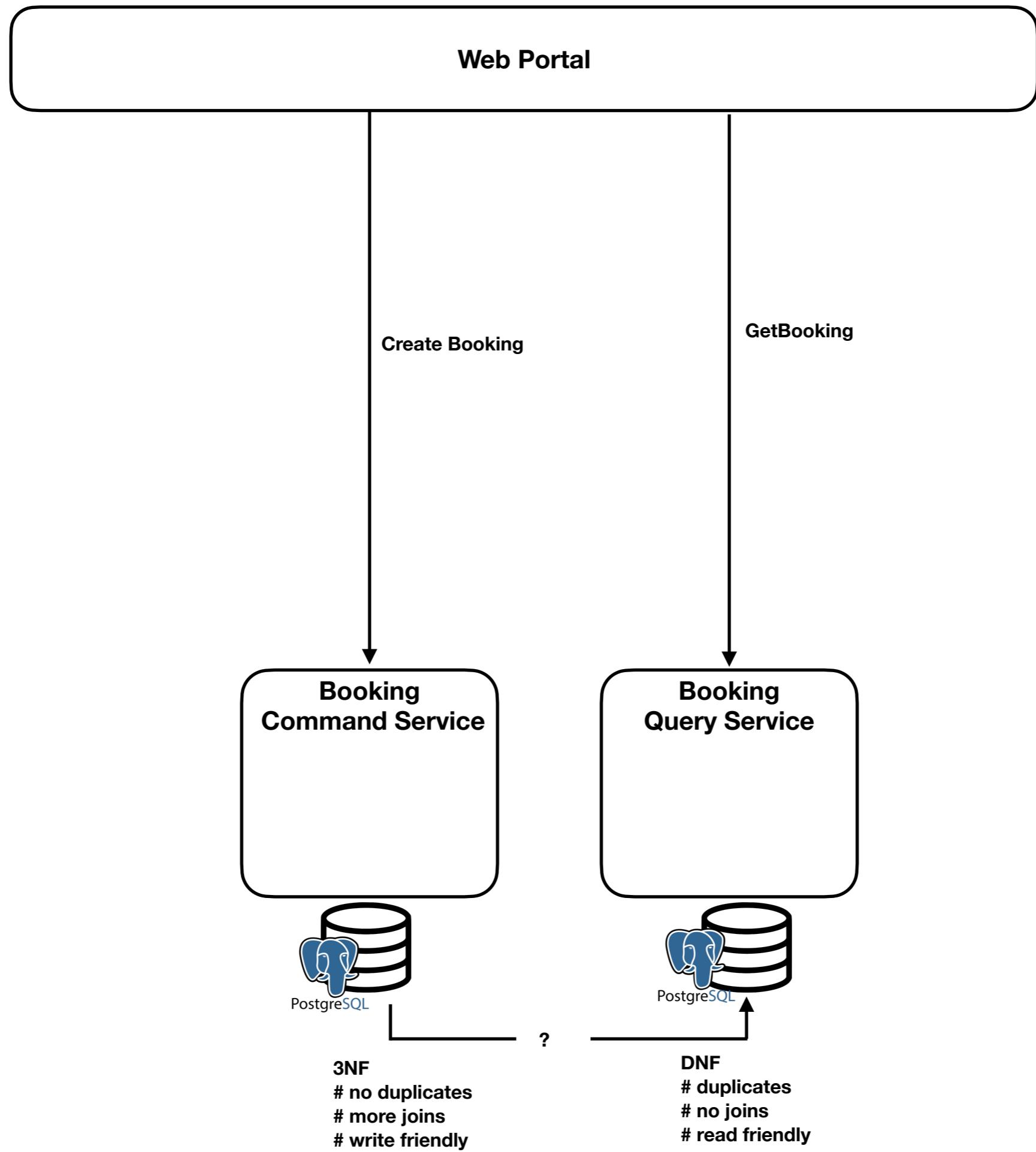


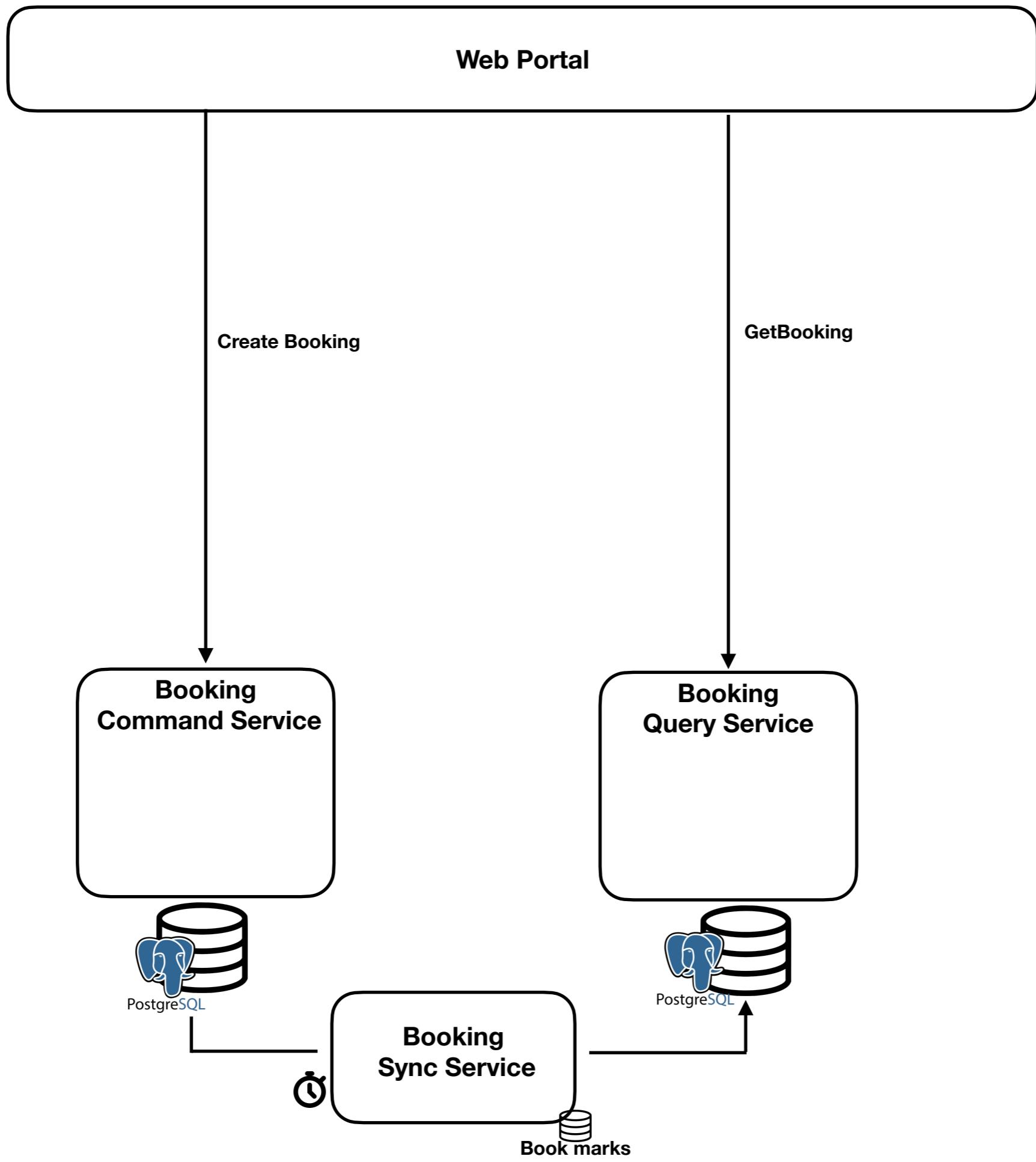


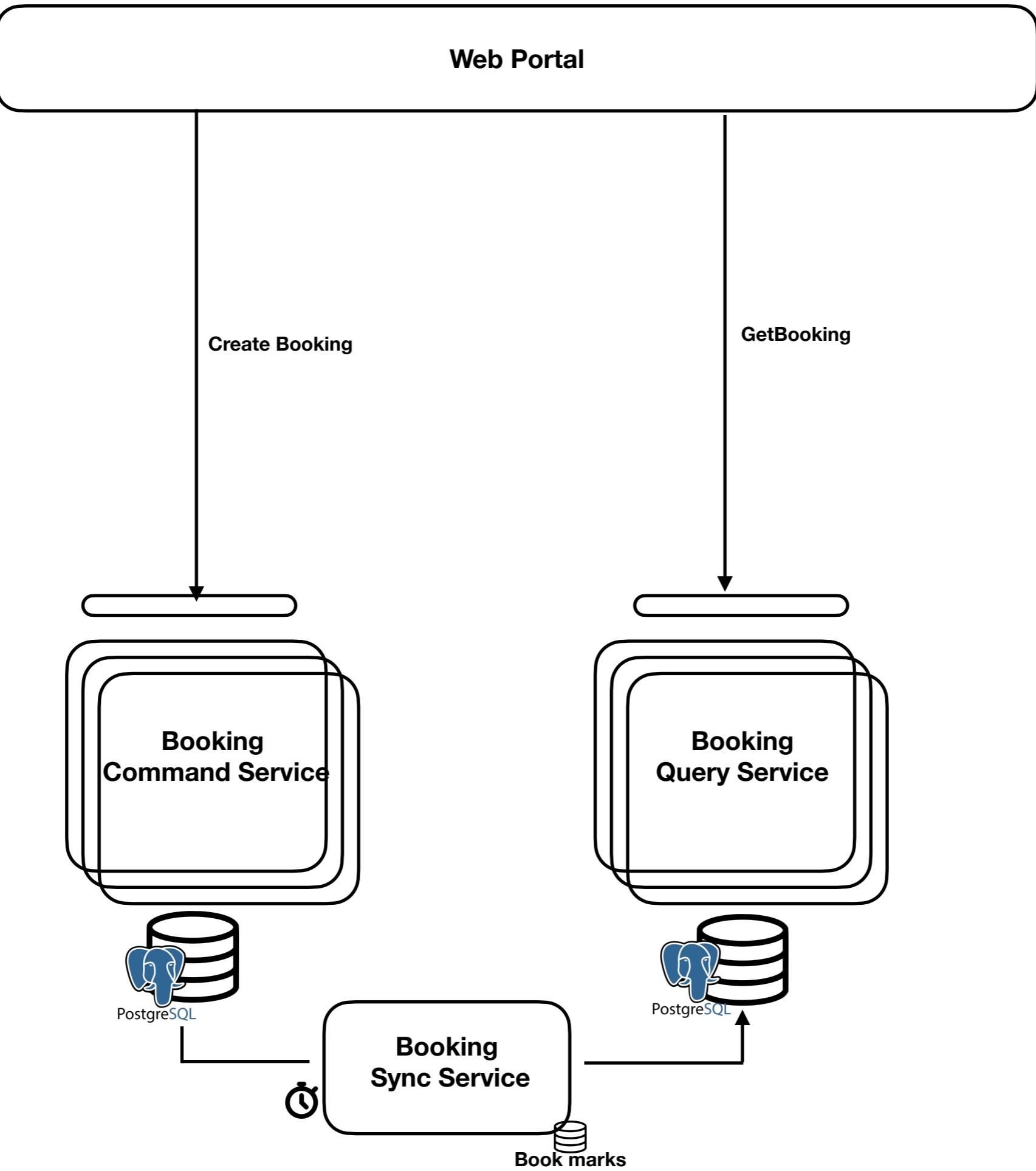


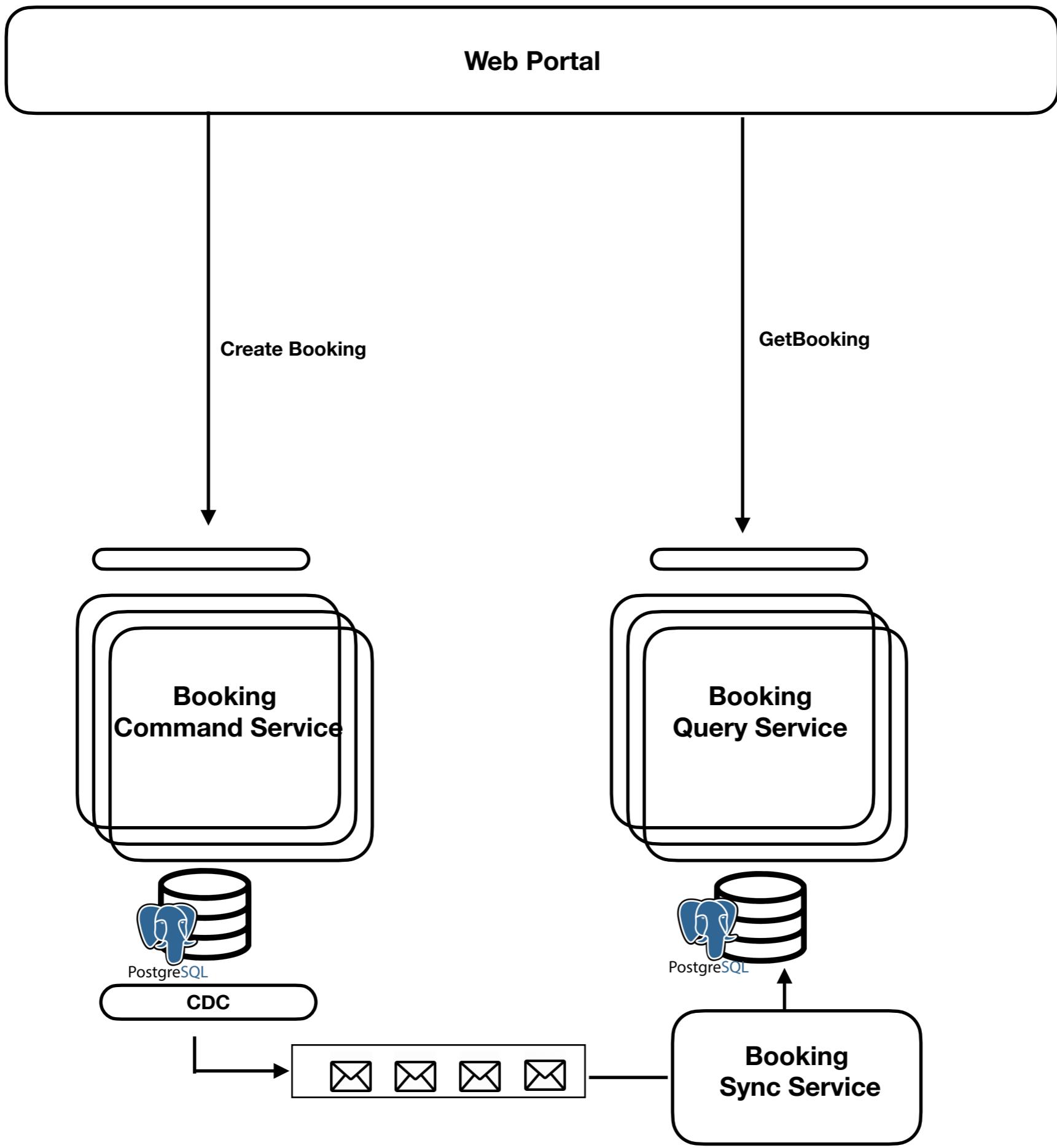
Service

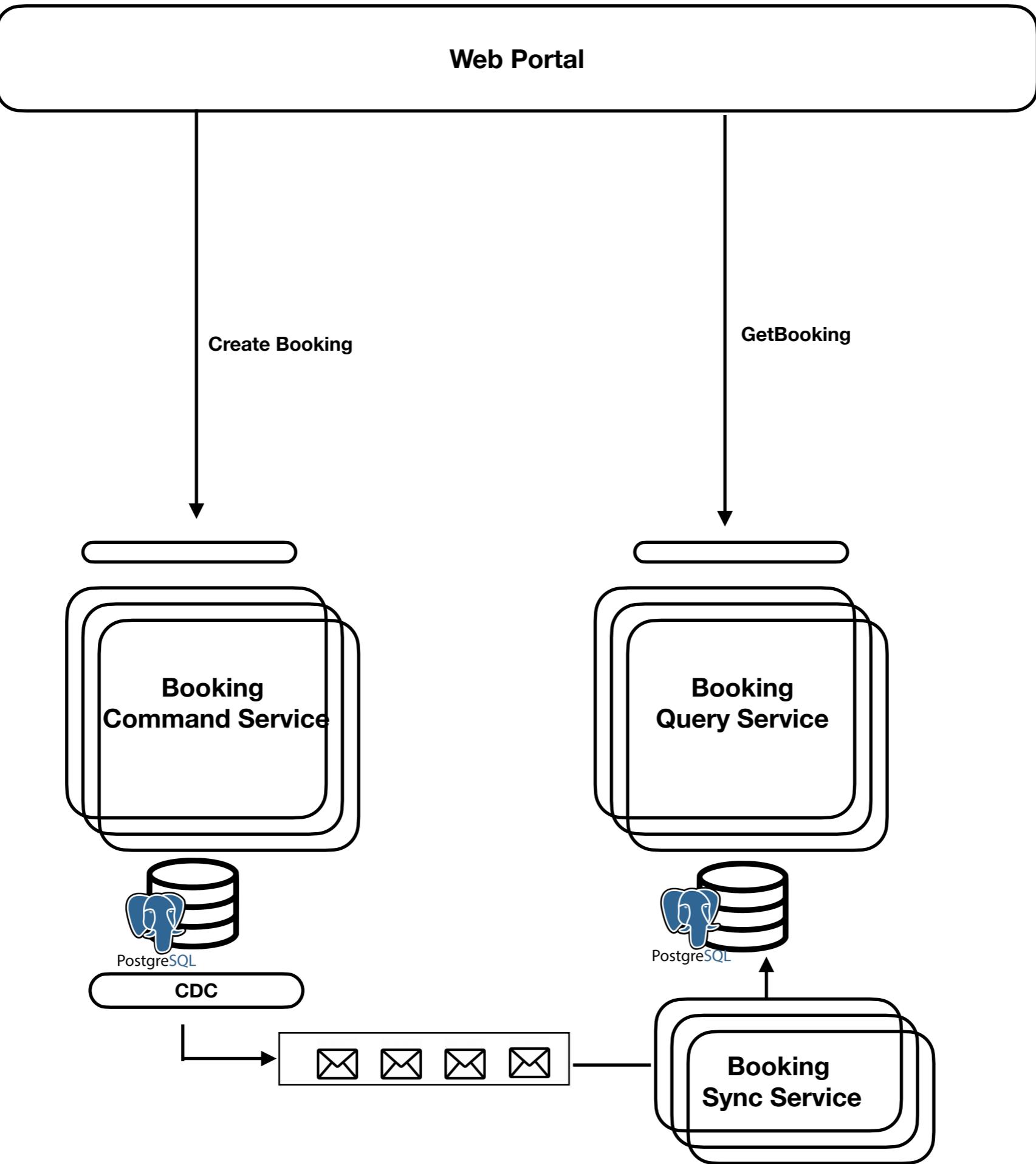
**# idempotency
unordered delivery
undoable operation**

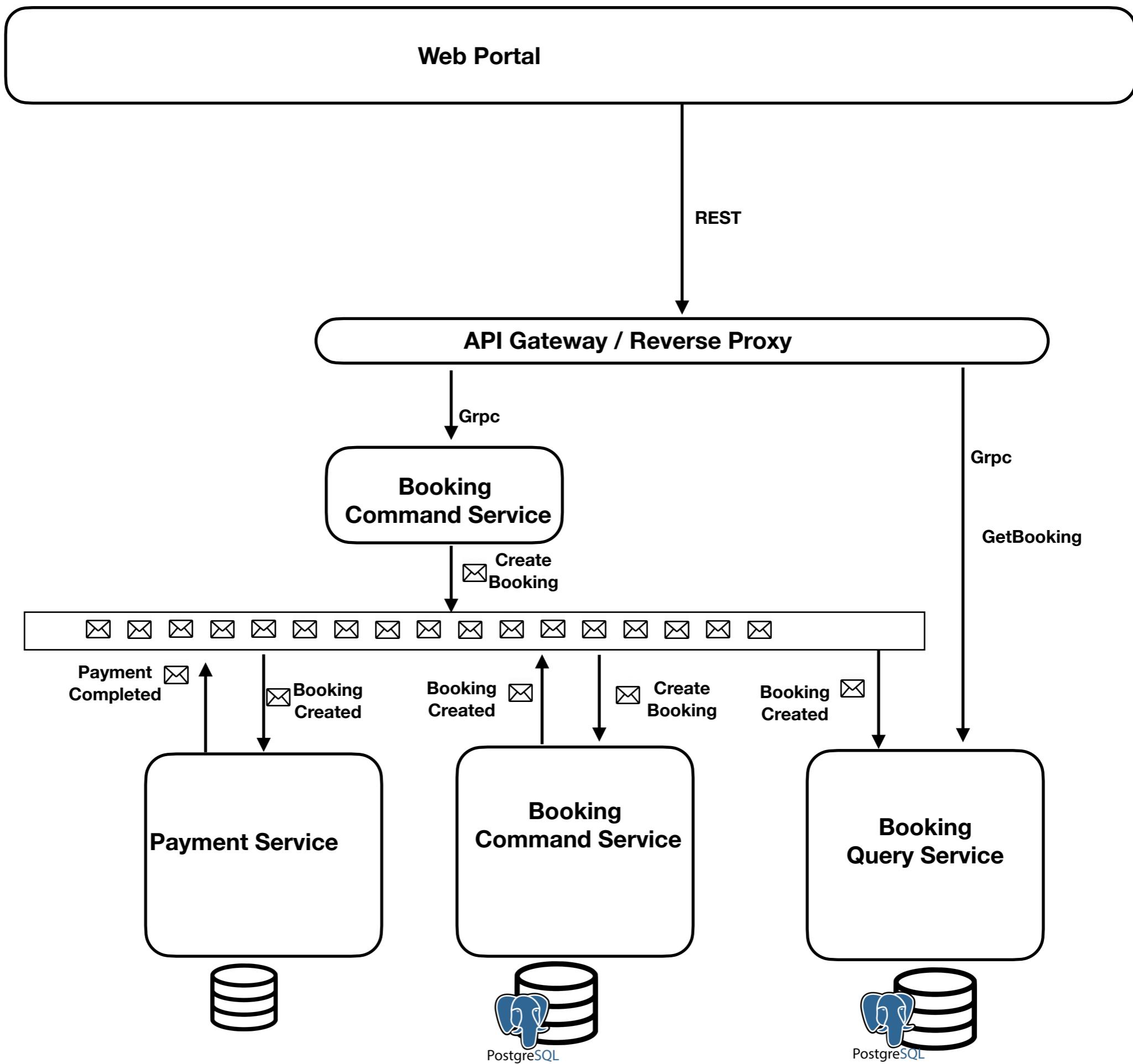




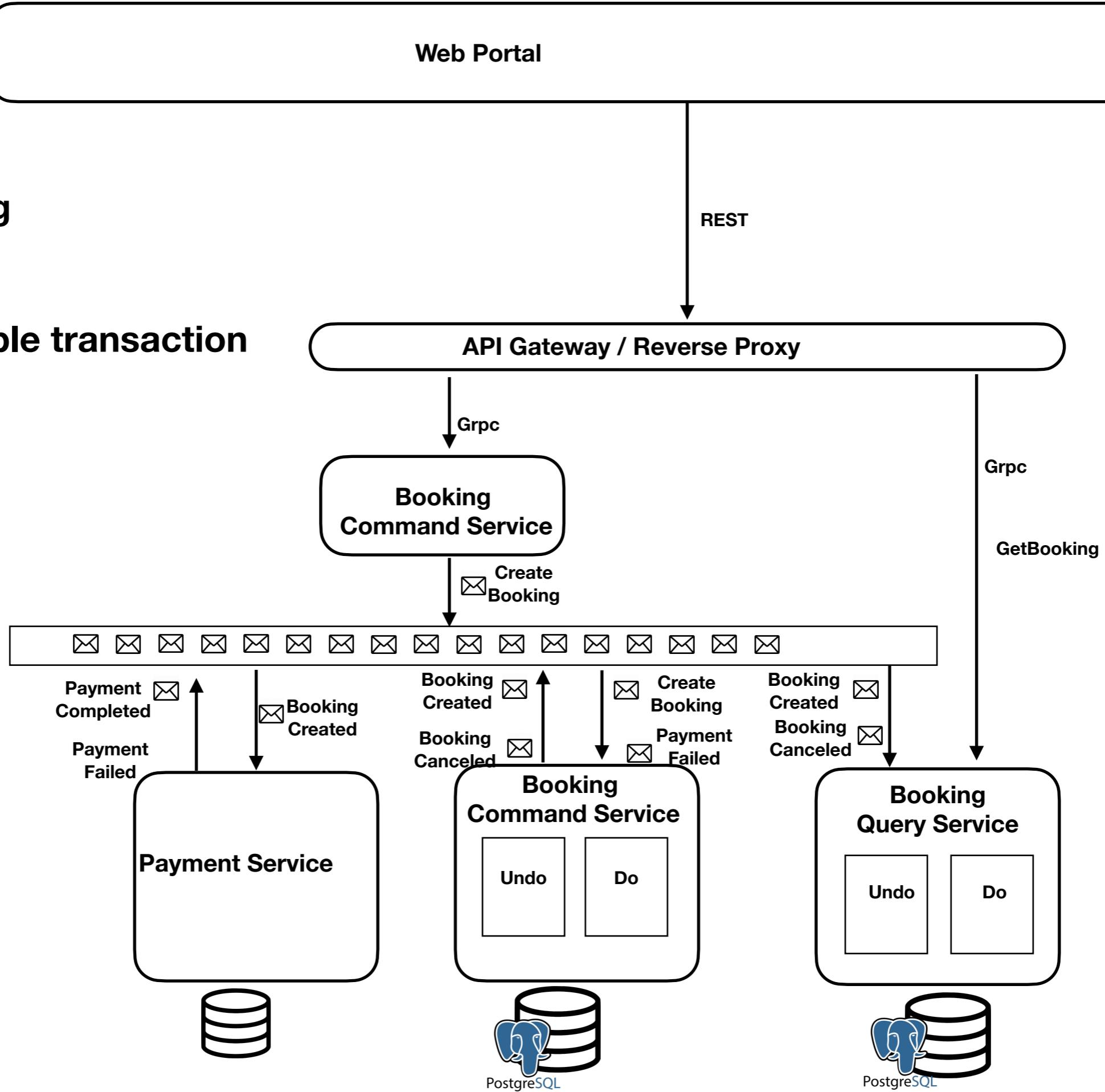


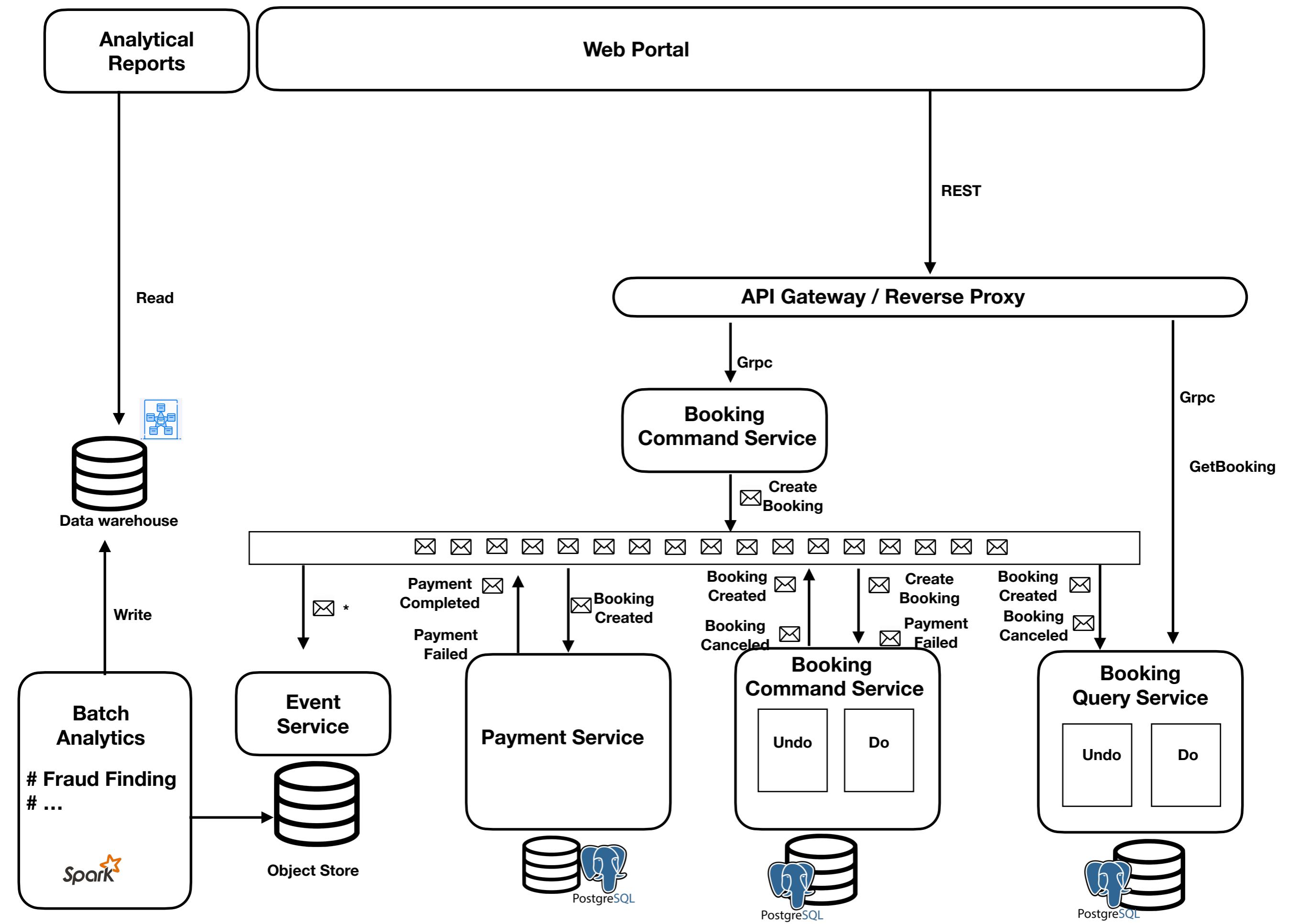






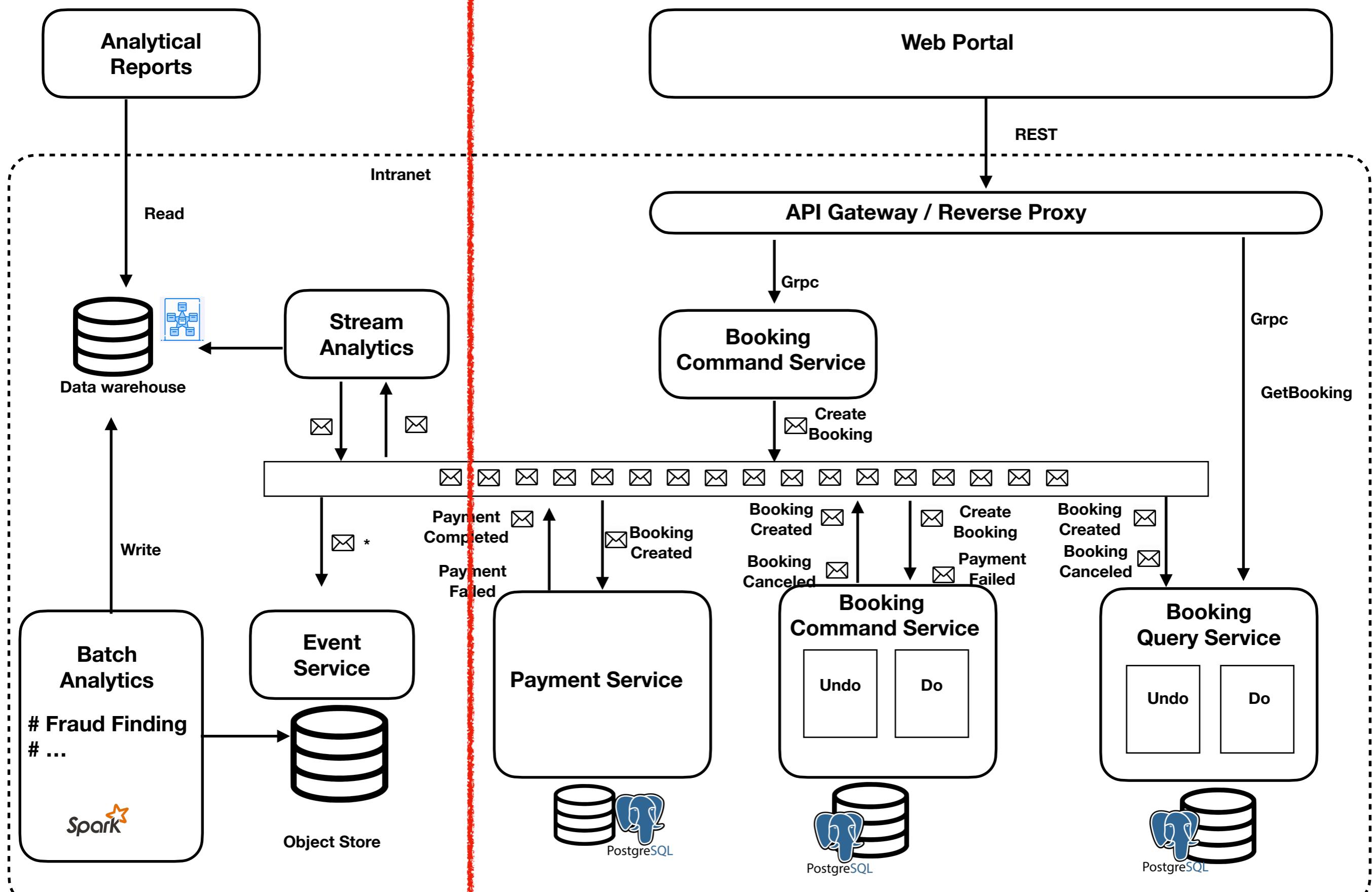
EDA
Event Sourcing
CQRS
SAGA pattern
- compensatable transaction





<<Manage>>

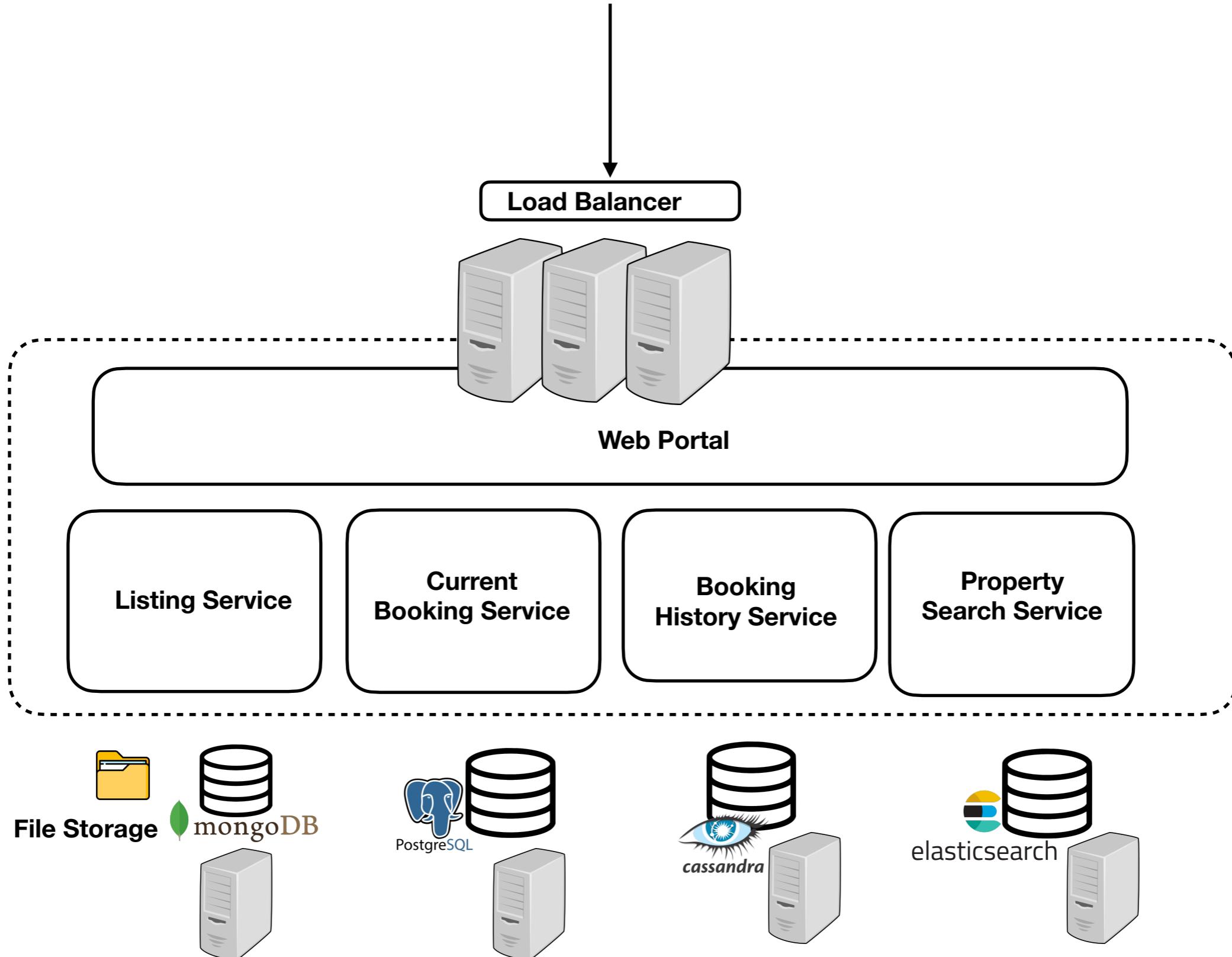
<<Run>>

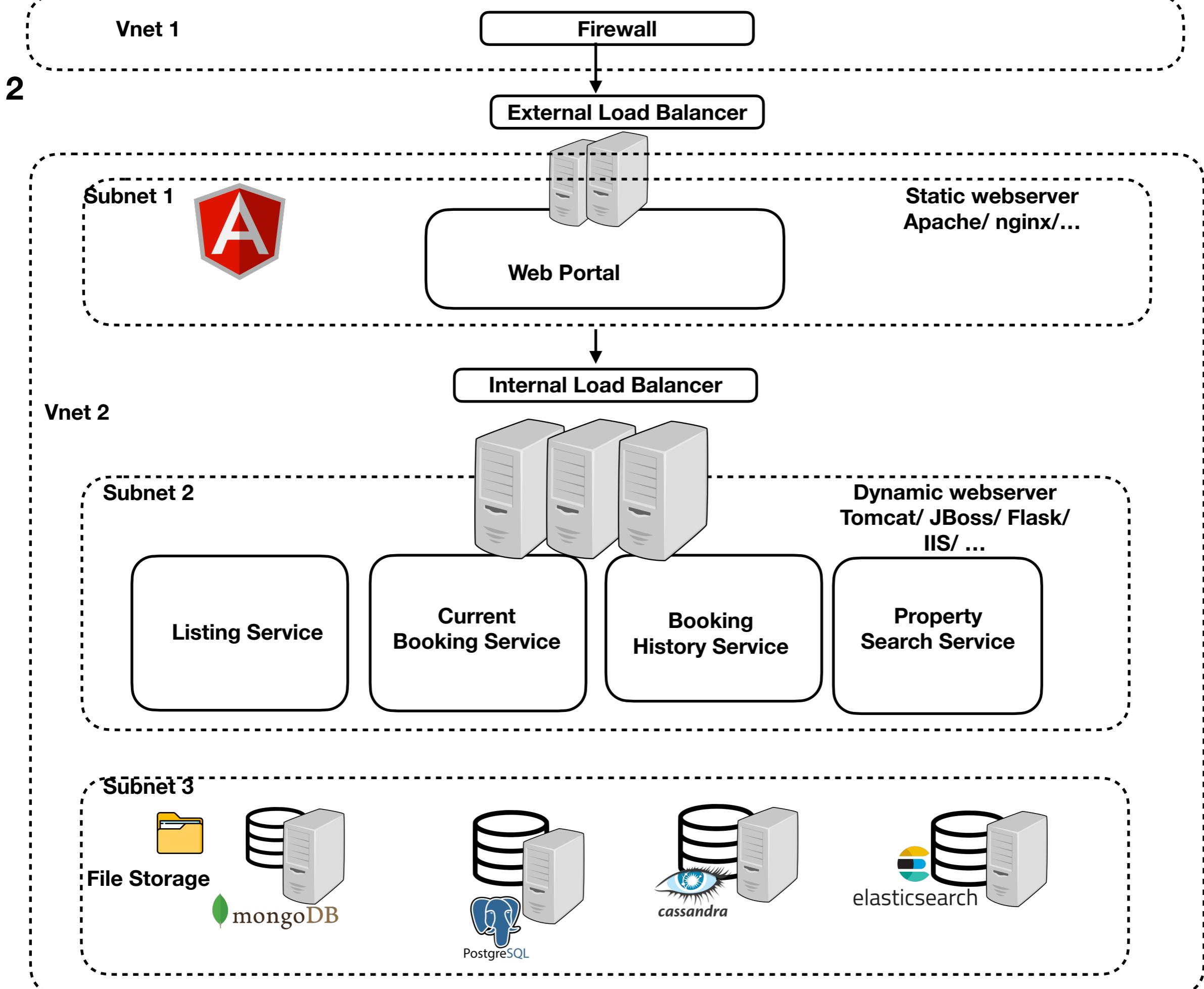


Deployment View

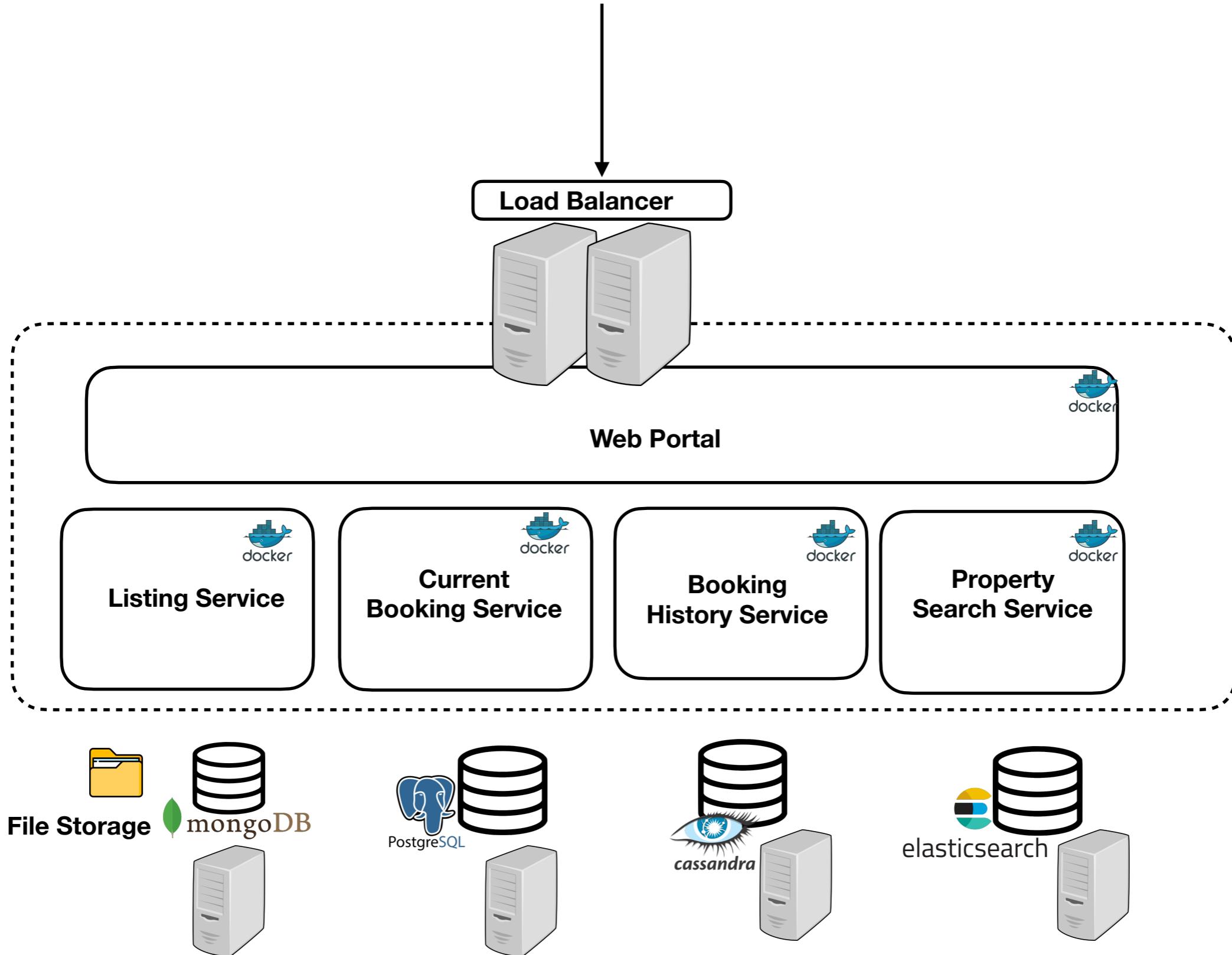
#

1

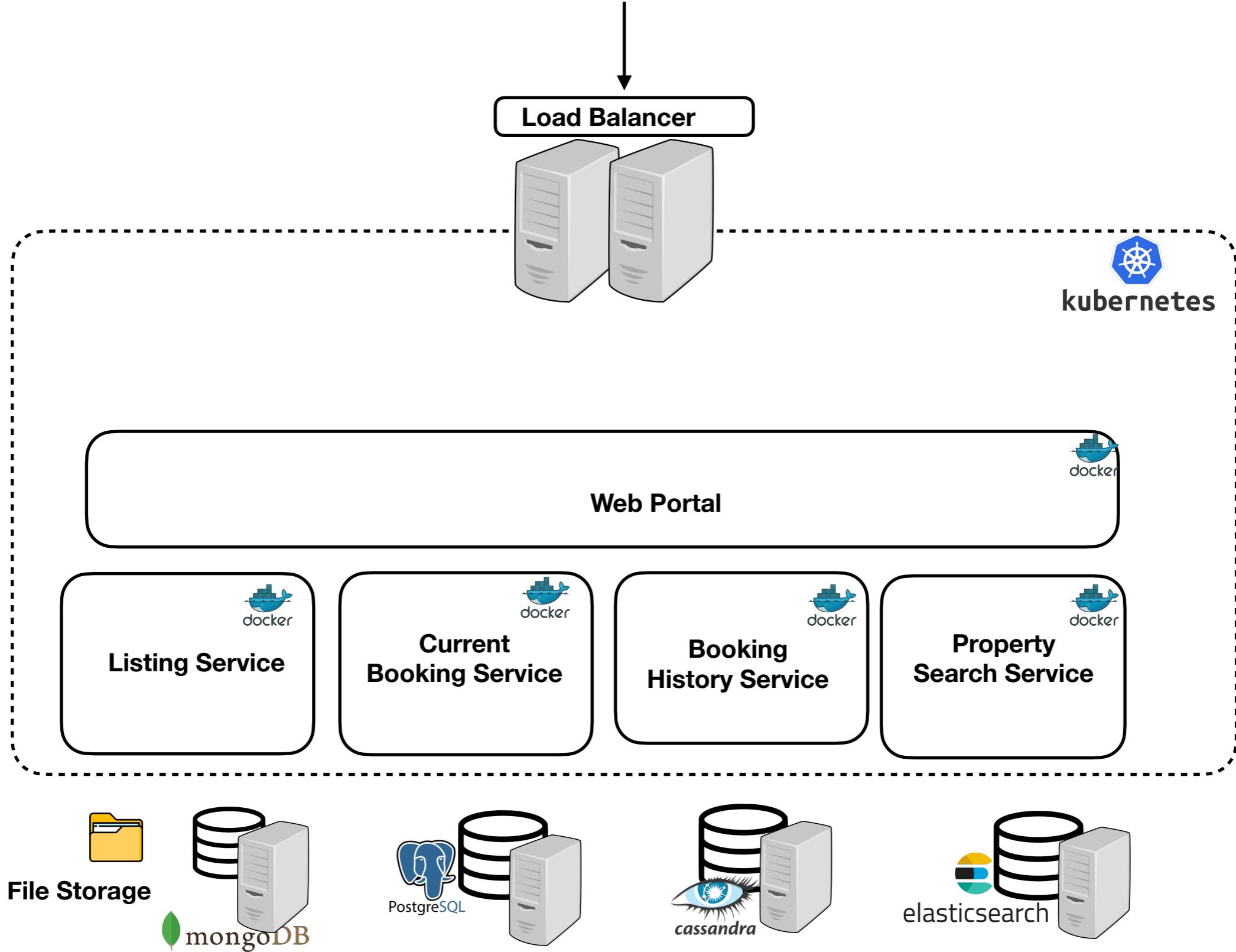




1



1



Architectural Justification

Evaluation (*ATAM | ARID | SAAM|)

1. identify all Architectural approaches

- Ad1 : Use Cloud object Storage for image and videos
- Ad2 : use document db for listing
- Ad3 : use rdbms for booking
- Ad4 : move booking history to a wide column data store and use rdbms only for current booking
- Ad5: use text search for property search
- Ad6 : decompose system by Domain (Property, booking, Search,...)
- Ad7 : decompose portal using interaction pattern
- Ad 8: decompose domain services using layered pattern
- Ad 9: use Managed K8s to deploy Domain services and portal
- Ad 10: use IAAS for database deployments
- Ad 11: use OpenID connect + OAuth2 for Authentication using Auth0
- Ad 12: use Claims to centralize Authorization (token + claim)
- Ad 13: use Event sourcing for Auditing
- Ad 14: use Event Driven Architecture to loosely couple micro services
- Ad 15: use WAF to validate HTTP
- Ad 16: Use CDN to cache image and videos in object storage
- Ad 17: Use API gateway to central validate Auth token and claims

2. identify all quality requirements

qs1. A guest Searching for a property On the web portal during peak hours should get property listing In < 2 seconds.

qs2. A unknown identity requests to add a property in the Web Portal during normal hours. The response is to block access to the data and record the access attempts with 100% probability of detecting the attack, 100% probability of denying access, and 50% probability of identifying the location of the individual.

qs3. A guest Submits a booking On the portal Duplicate submit The guest is not doubly charged With a 100% probability of detecting the duplicate request.

qs4. Developer Want to add a new payment gateway On the portal During maintenance The payment gateway is added In < 2 man days.

qs5. The Database Failed In the Data Centre during Operational Hours Secondary is made Primary In < 2 minutes.

3. analyse Scenario -> Approach

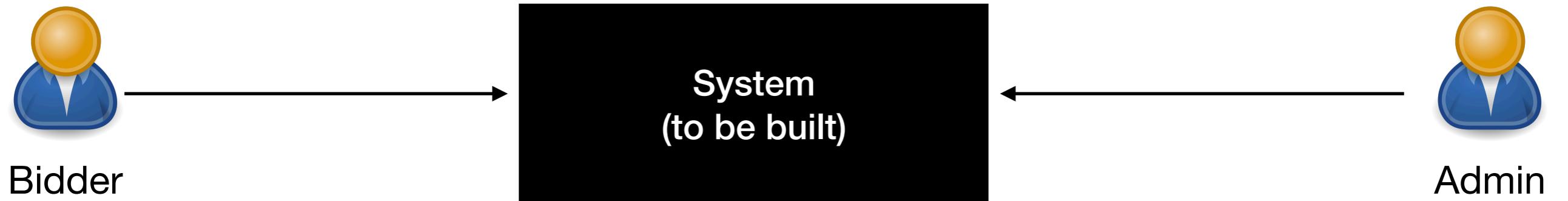
Quality Scenario	Approach	Risk	Remarks
Qs1	1,2,5,6,9,16	Low	Can think of caching the properties
Qs2	11, 12, 13, 15, 17	Med,	L 3 Network firewall , IP location identification service ?
Qs3	?	High	Booking service should be designed for idempotency
QS4	Ad6, Ad9	Med	Design for pluggability
QS5	Ad9, Ad10	Low	

Bidding Case Study

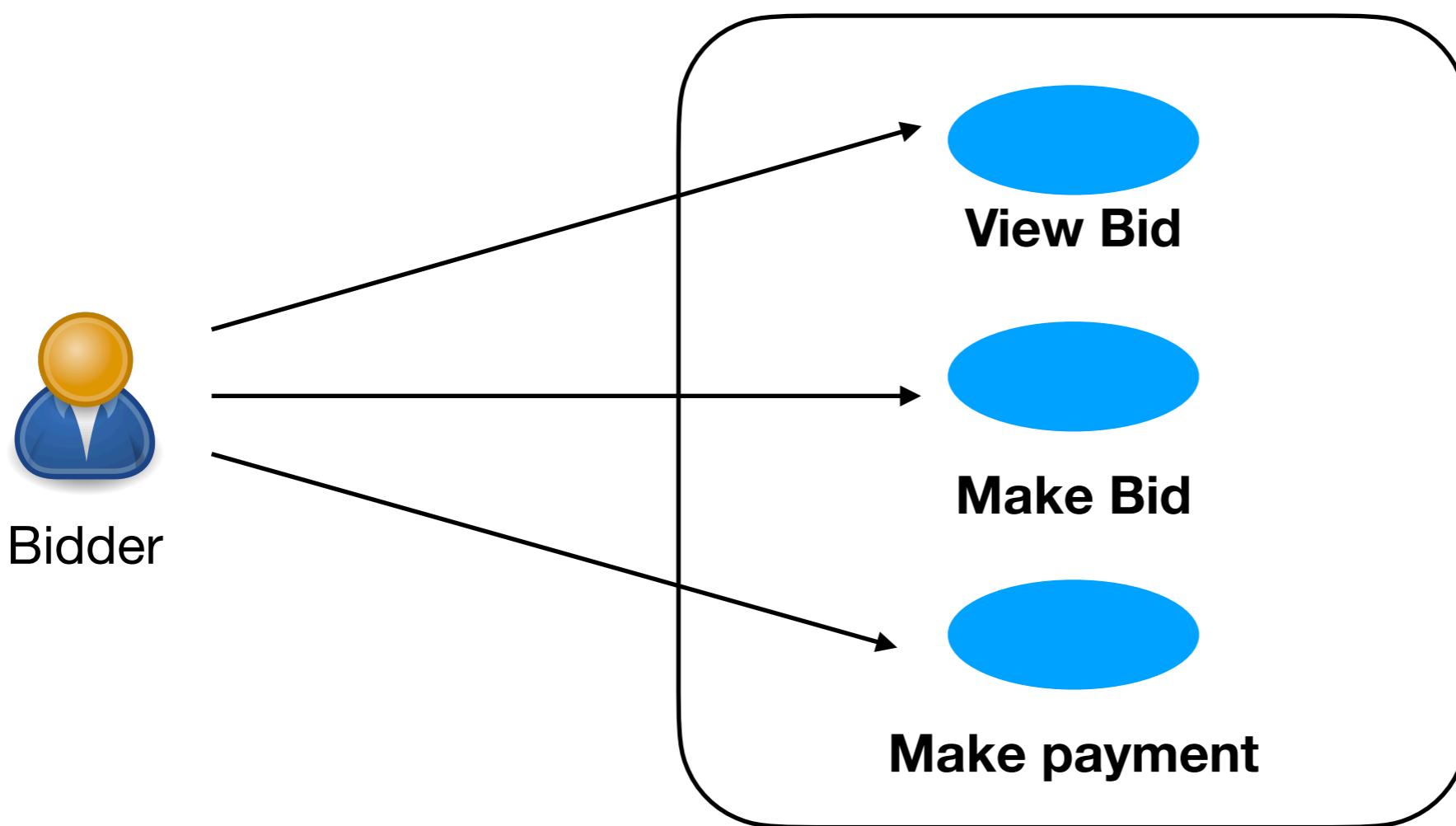
Architectural Requirements

- # Context View
- # Functional View
- # Constraints (removed)
- # Quality View *
- # Assumptions

Context View



Functional view



Quality View

Quality	Source (who)	Stimulus (action)	Artifact (which)	Environment (context)	Response (output)	Measure (scale)
Performance	As a Bidder	I should be able to see the Bids placed by other bidders	In the portal	When there are 100,000 bidders bidding	The Highest bid is shown to the bidder	In < 1 sec

Assumptions

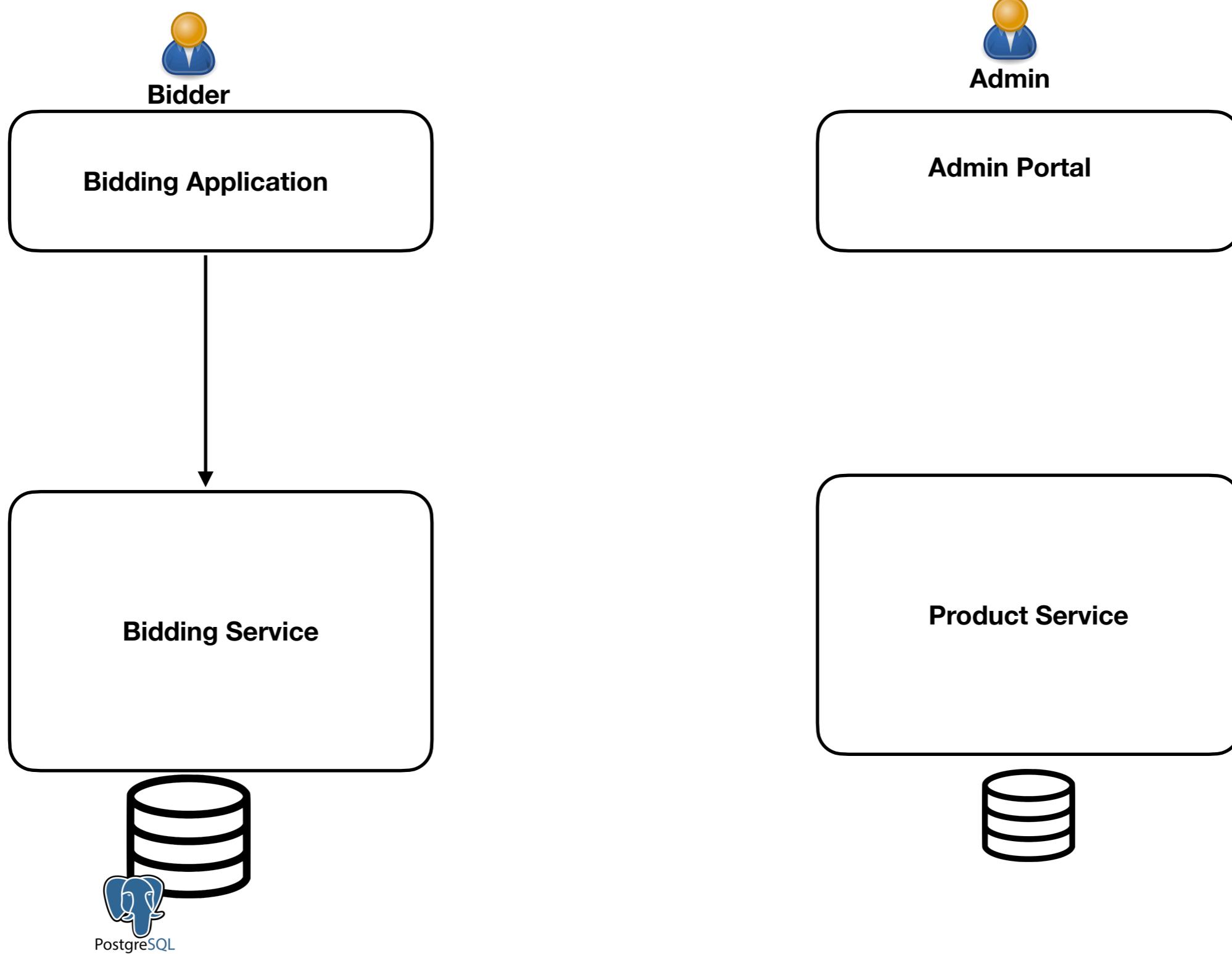
- User will only use Mobile Application. There is no web interface for the application.
- maximum bidding during peak load will be less than 100000 bids.
-

Architectural Design

- # Logical View
- # Deployment View
- # Security View (removed)

Logical View

- # System Decomposition**
- # Persistence approach**
- # Compute approach**
- # Communication approach**
- # cross cutting approach**



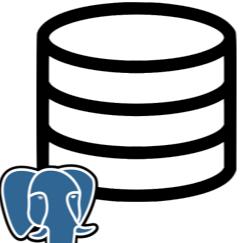


Bidder

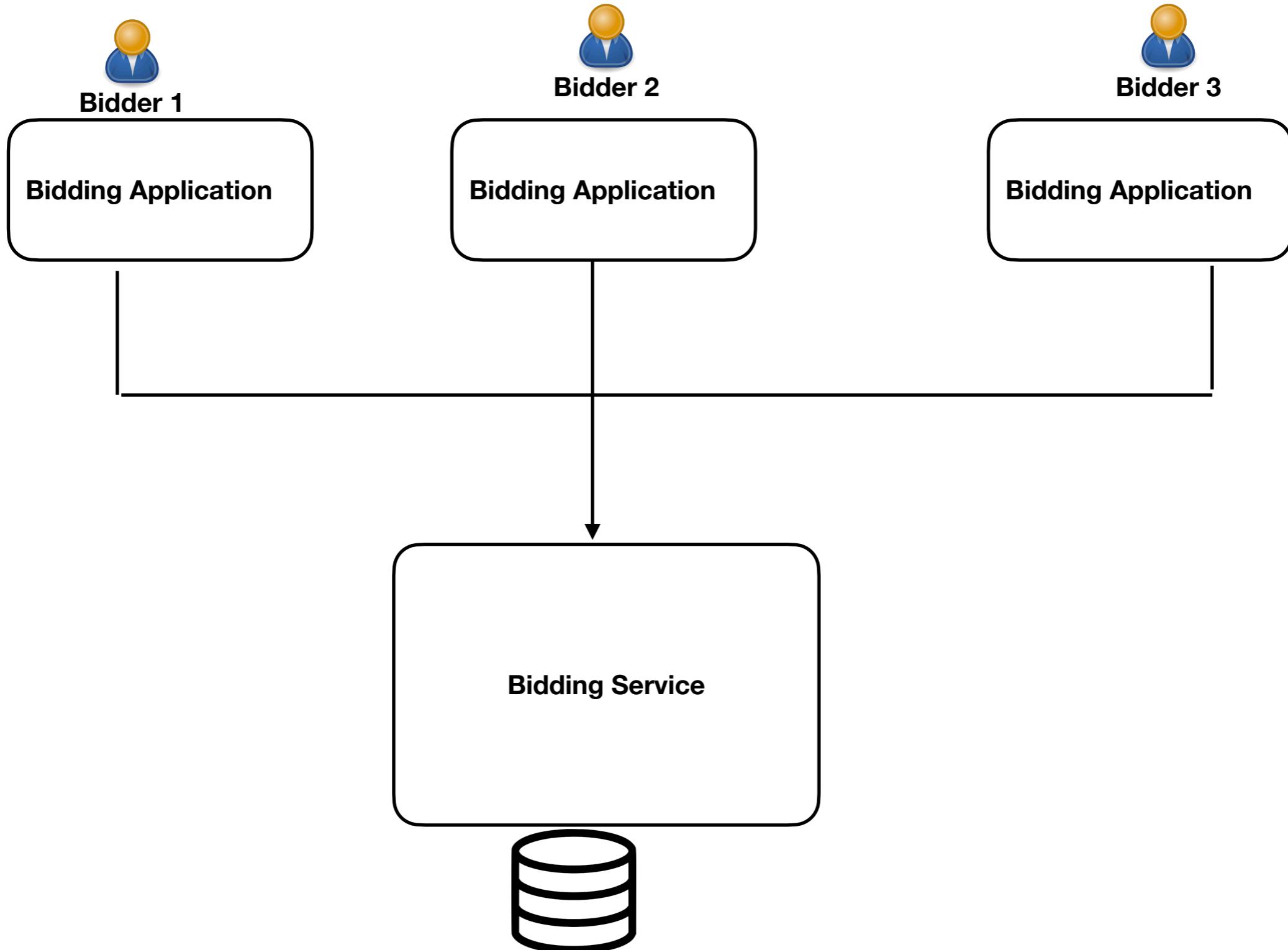
Bidding Application

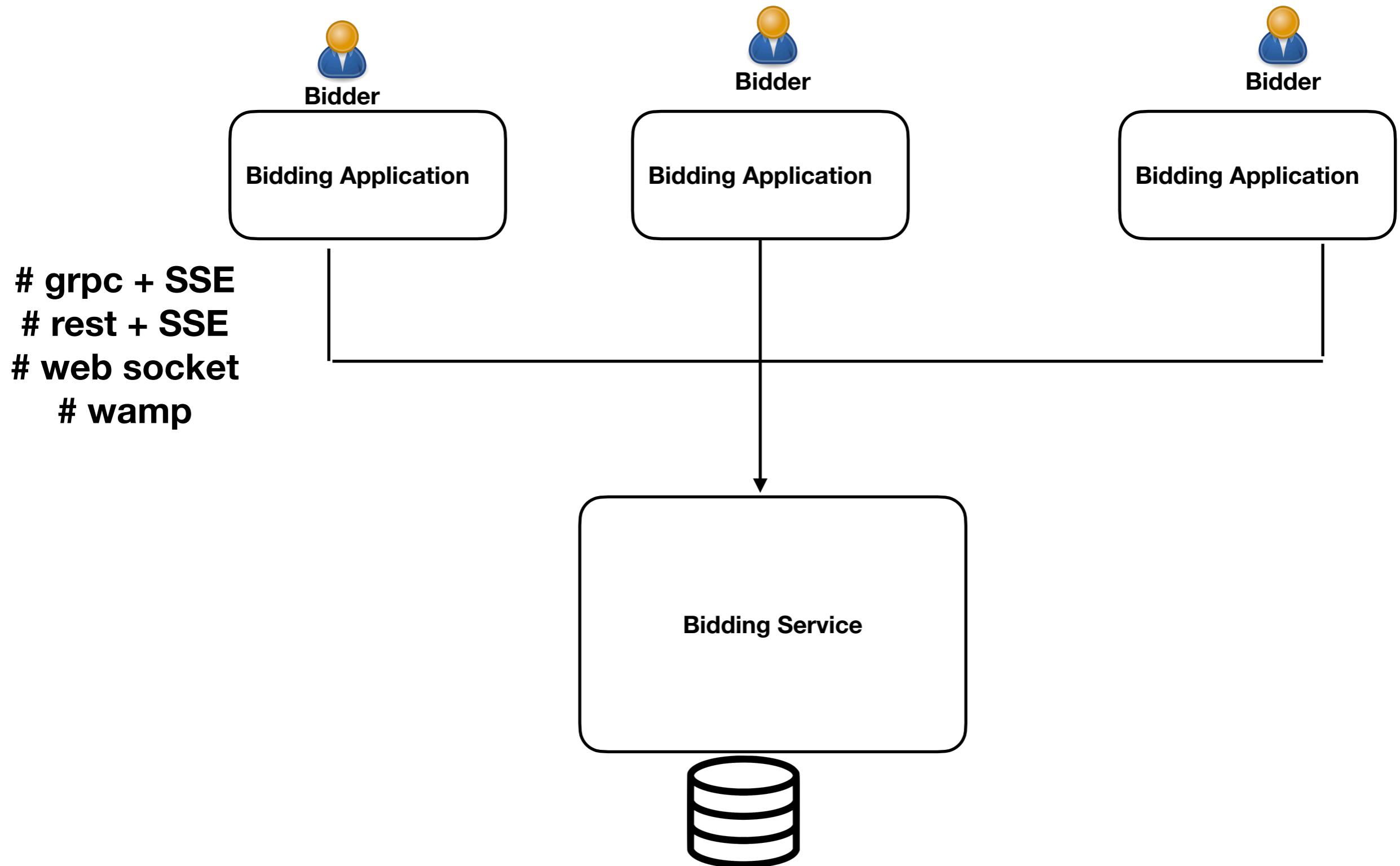


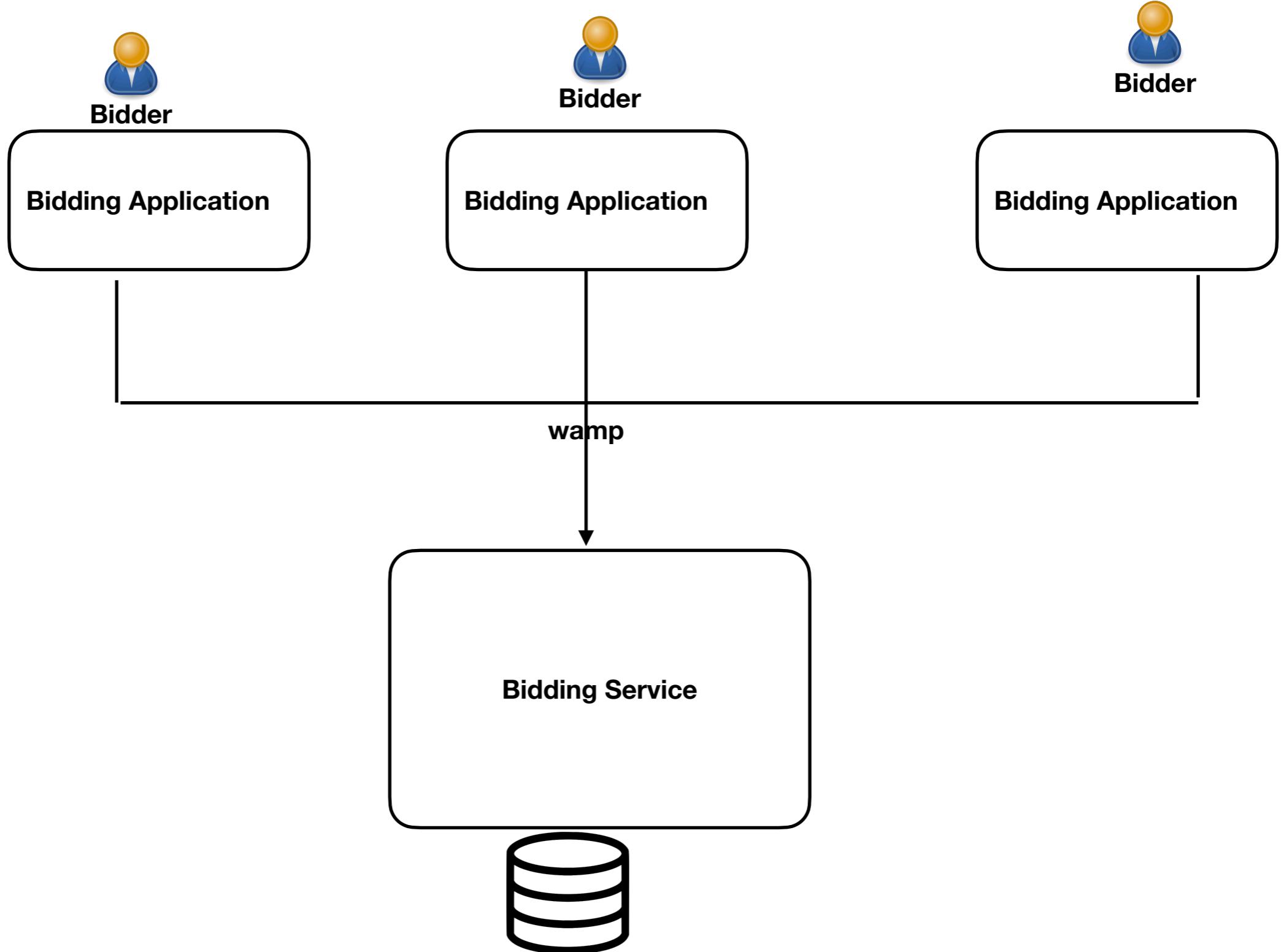
Bidding Service

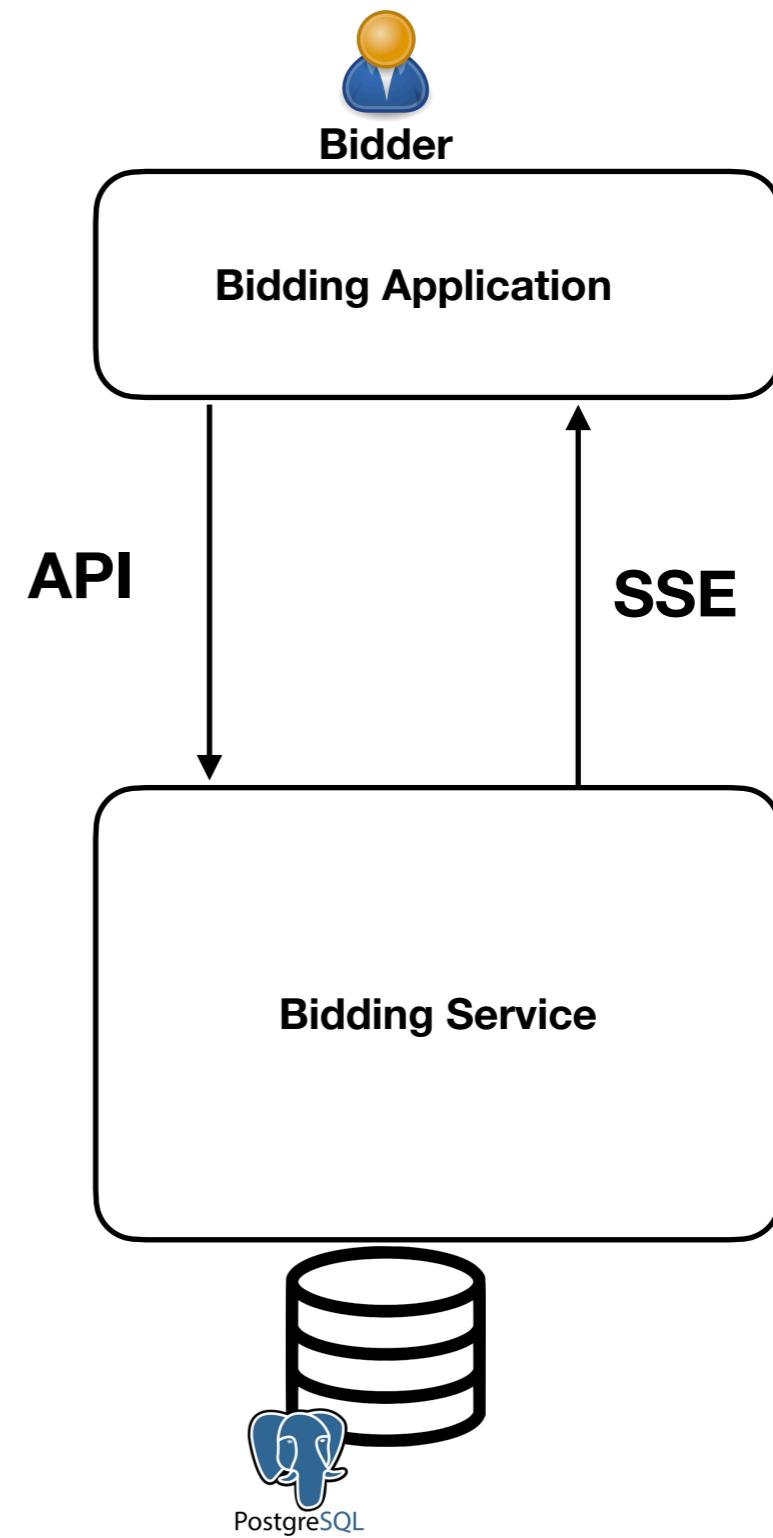
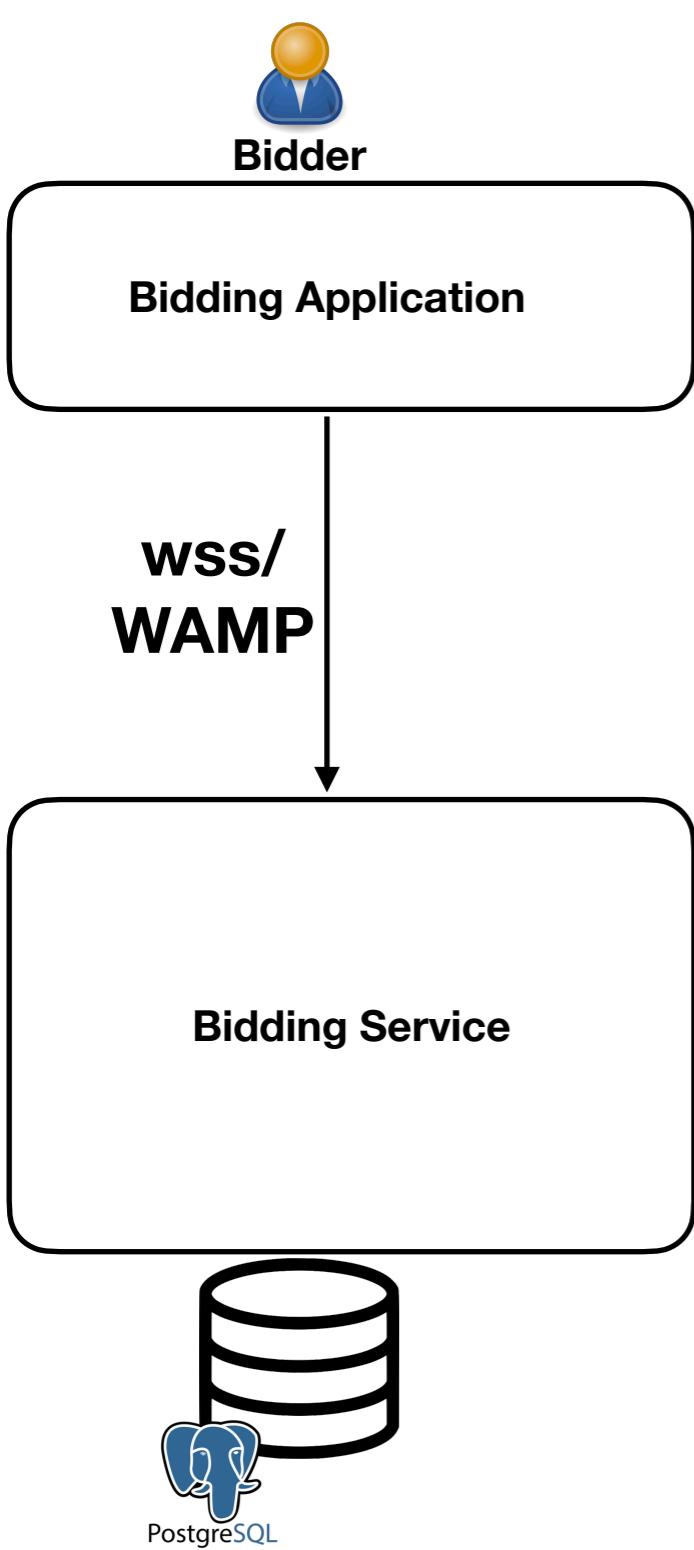


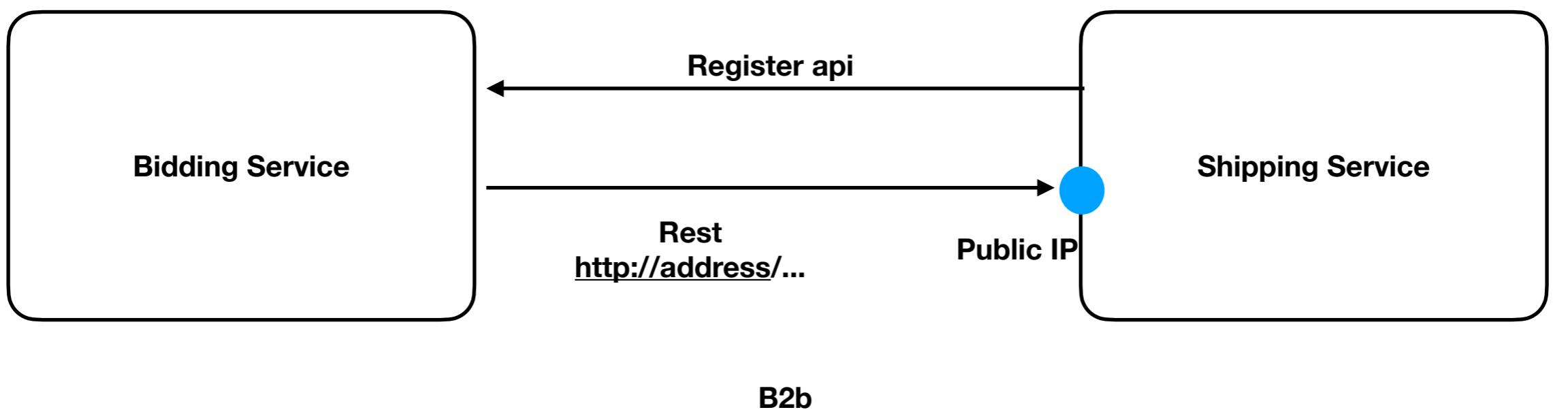
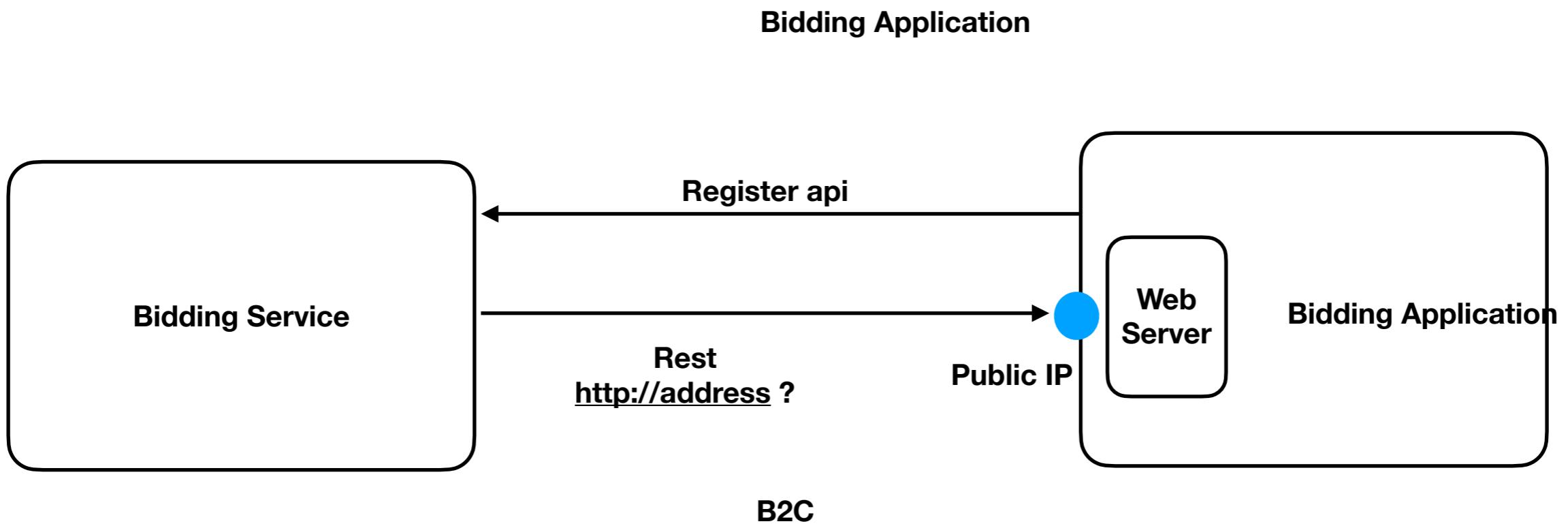
- Throttling -> DOS
- Https -> information disclosure
- AAA
 - Authentication
 - Authorization
 - Audit
- Input validation -> xss, injection, csrf, ...
- Output Encoding
- Zone Separations (DMZ , application, database) subnets
- Firewall and load balancer rules
-





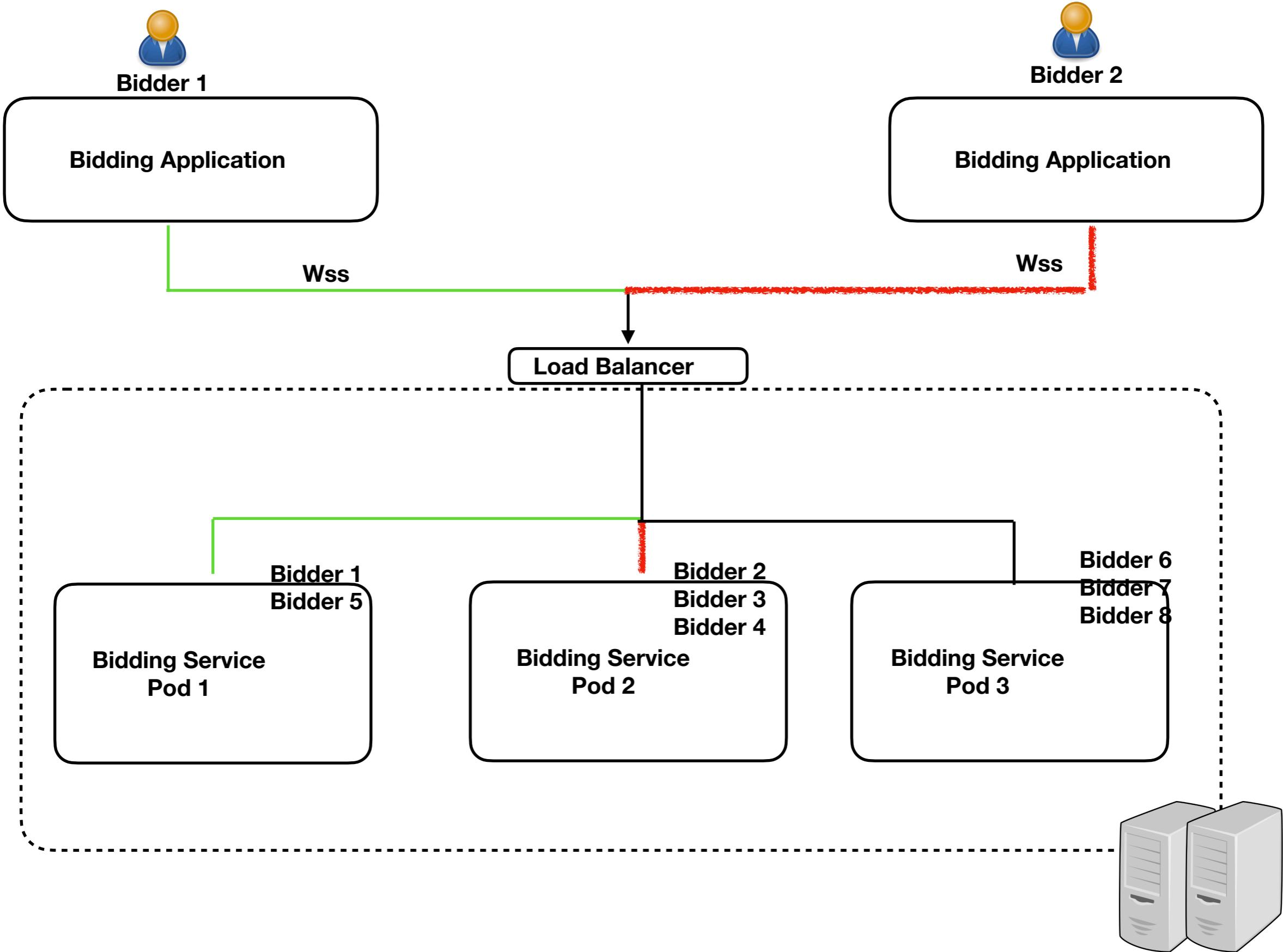






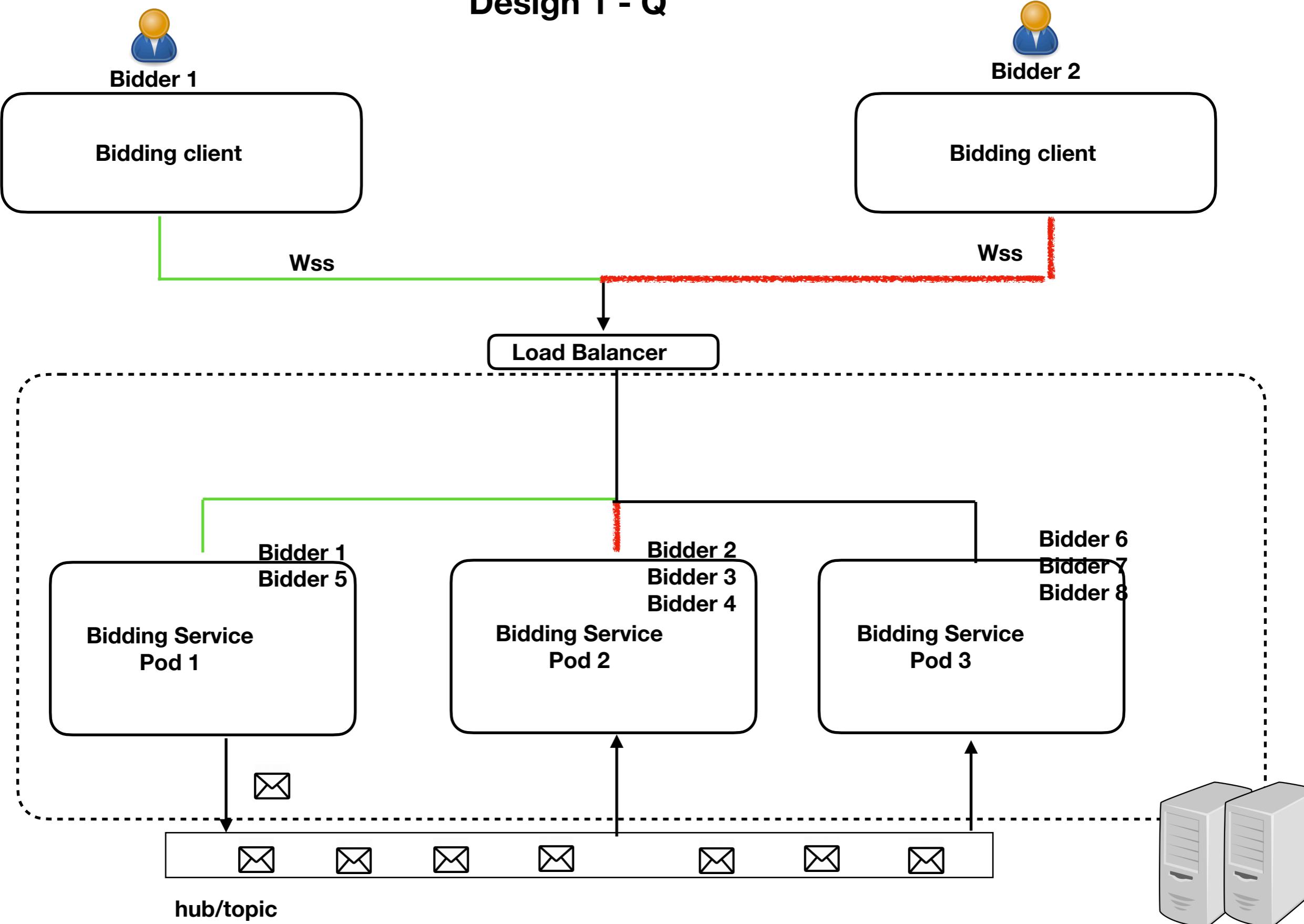
Deployment View

#

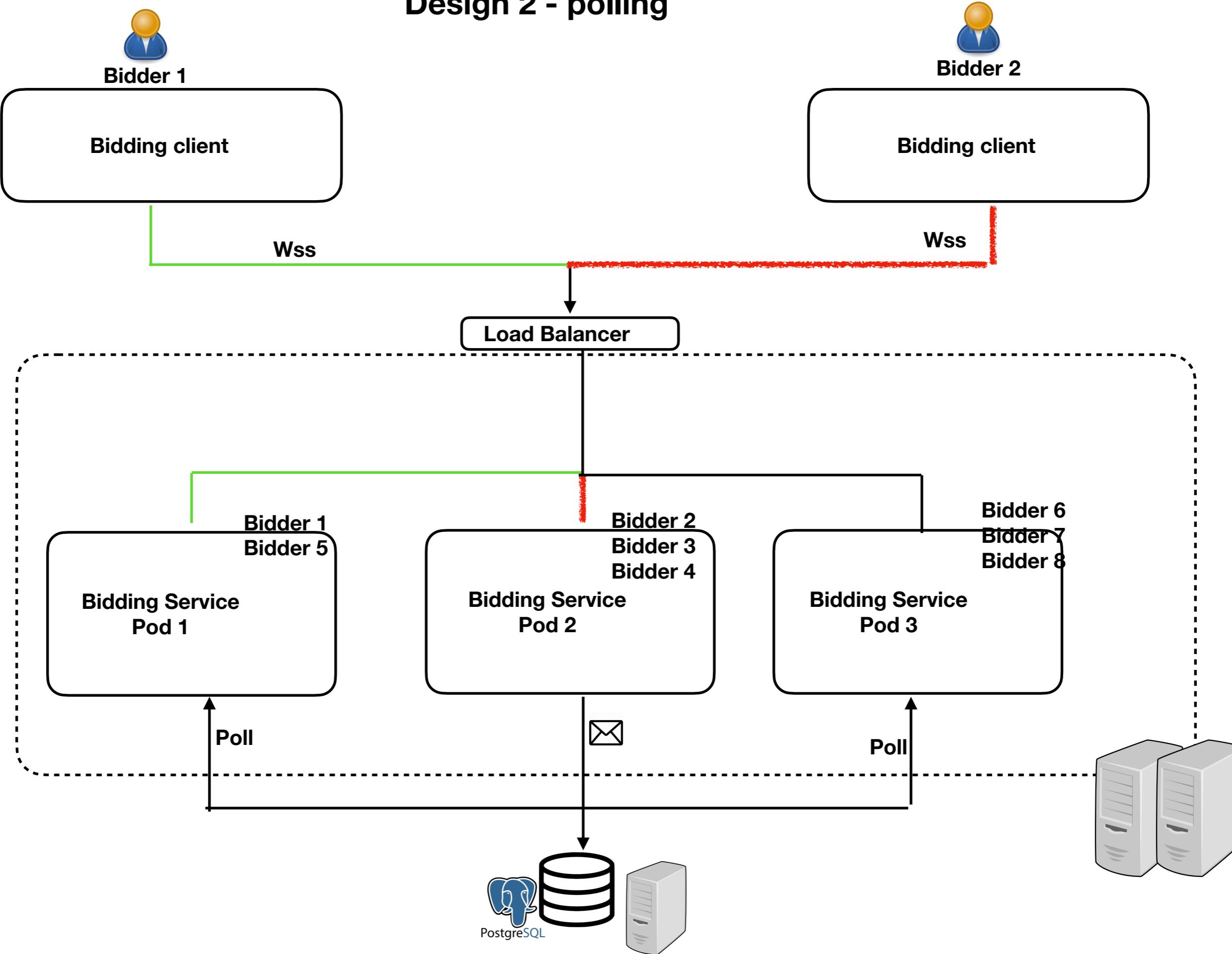


**ws vs api
scalability**

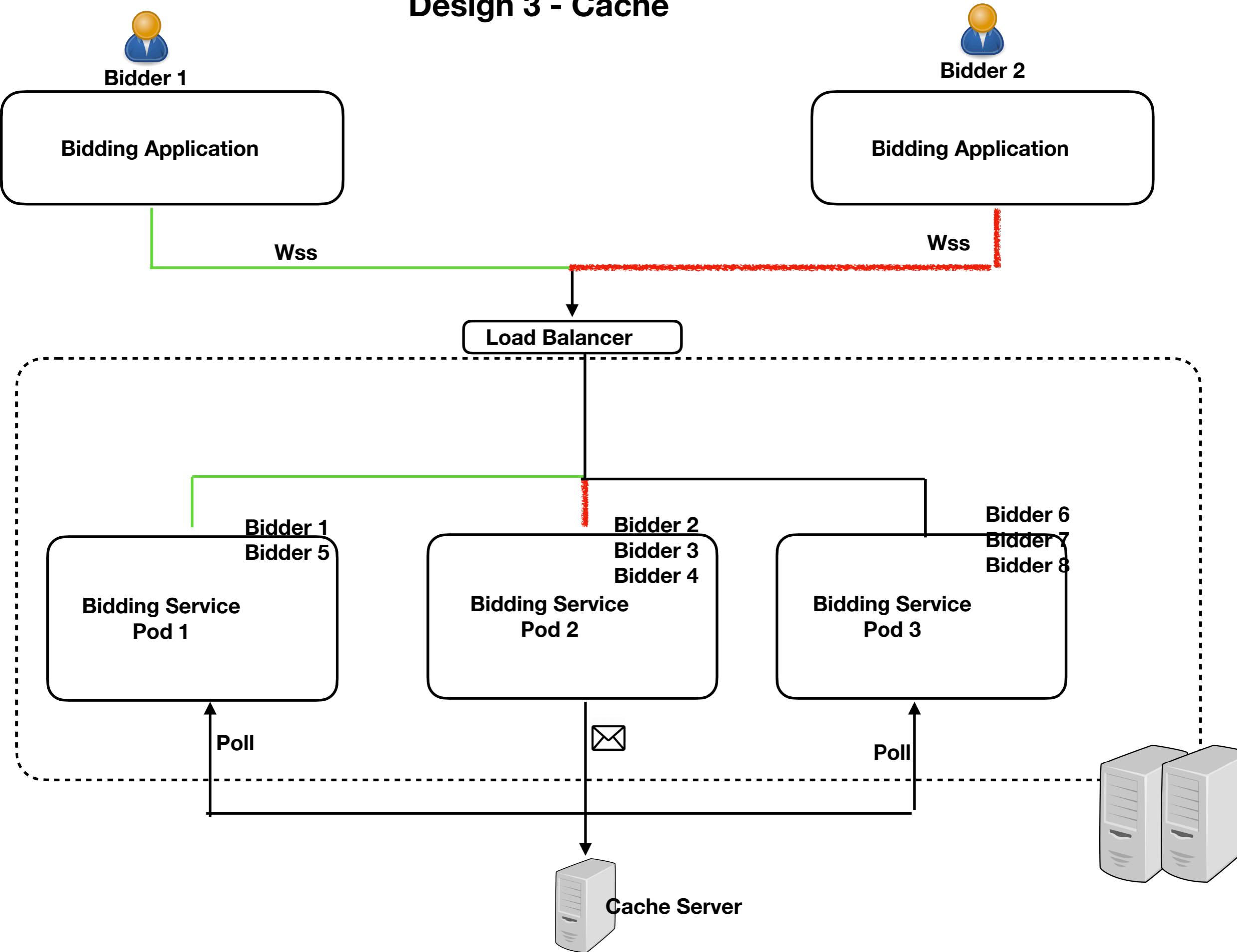
Design 1 - Q



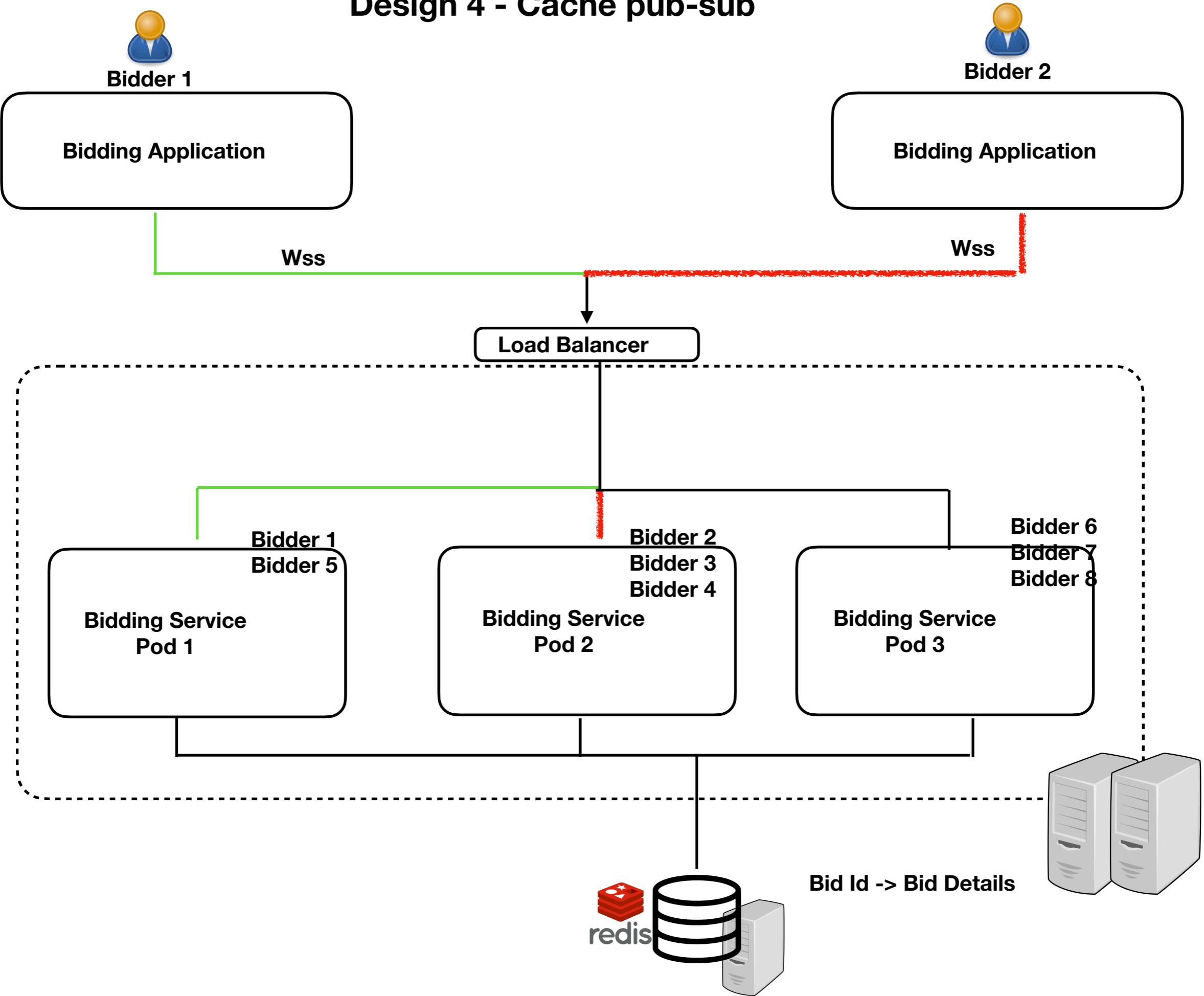
Design 2 - polling



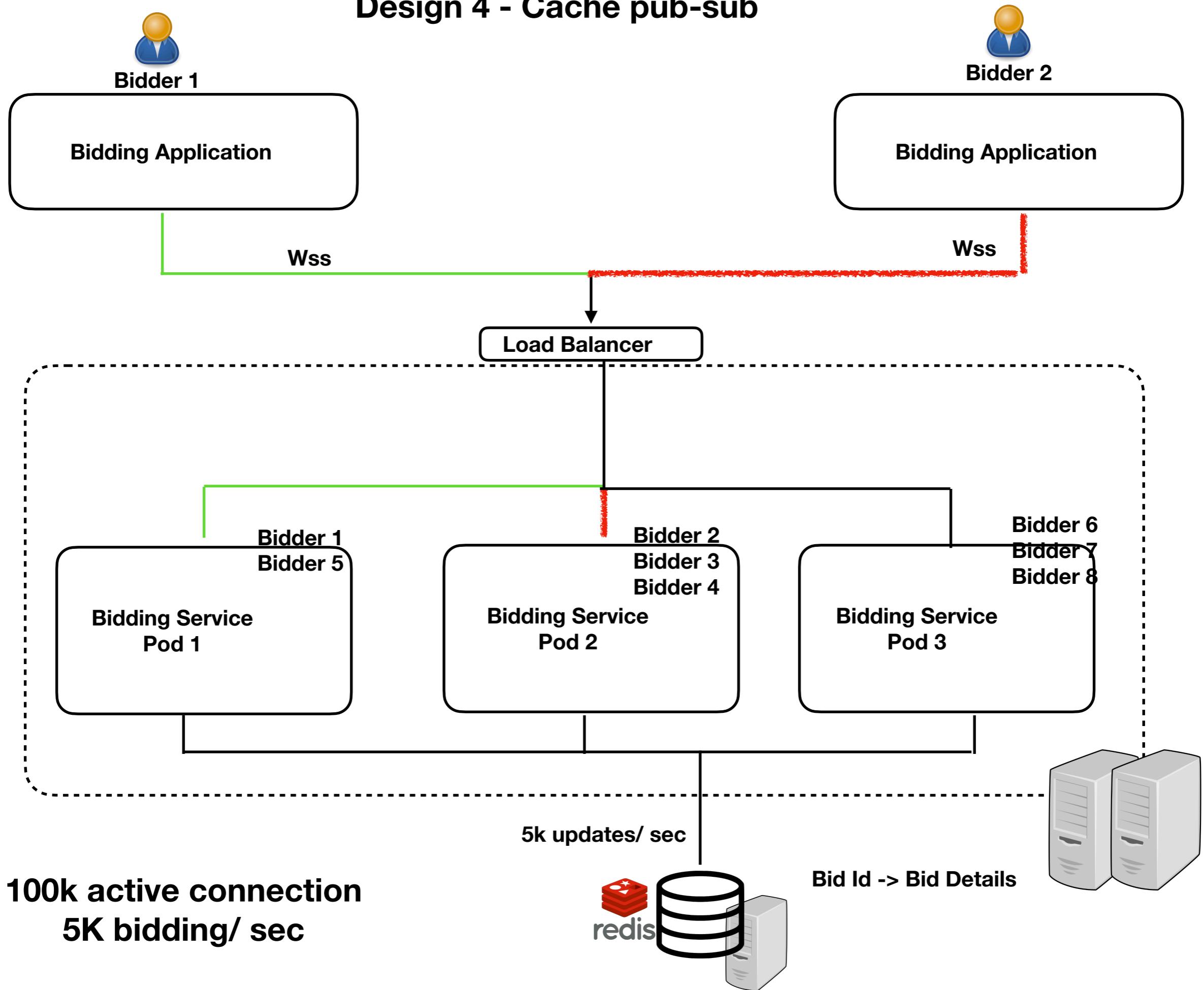
Design 3 - Cache



Design 4 - Cache pub-sub

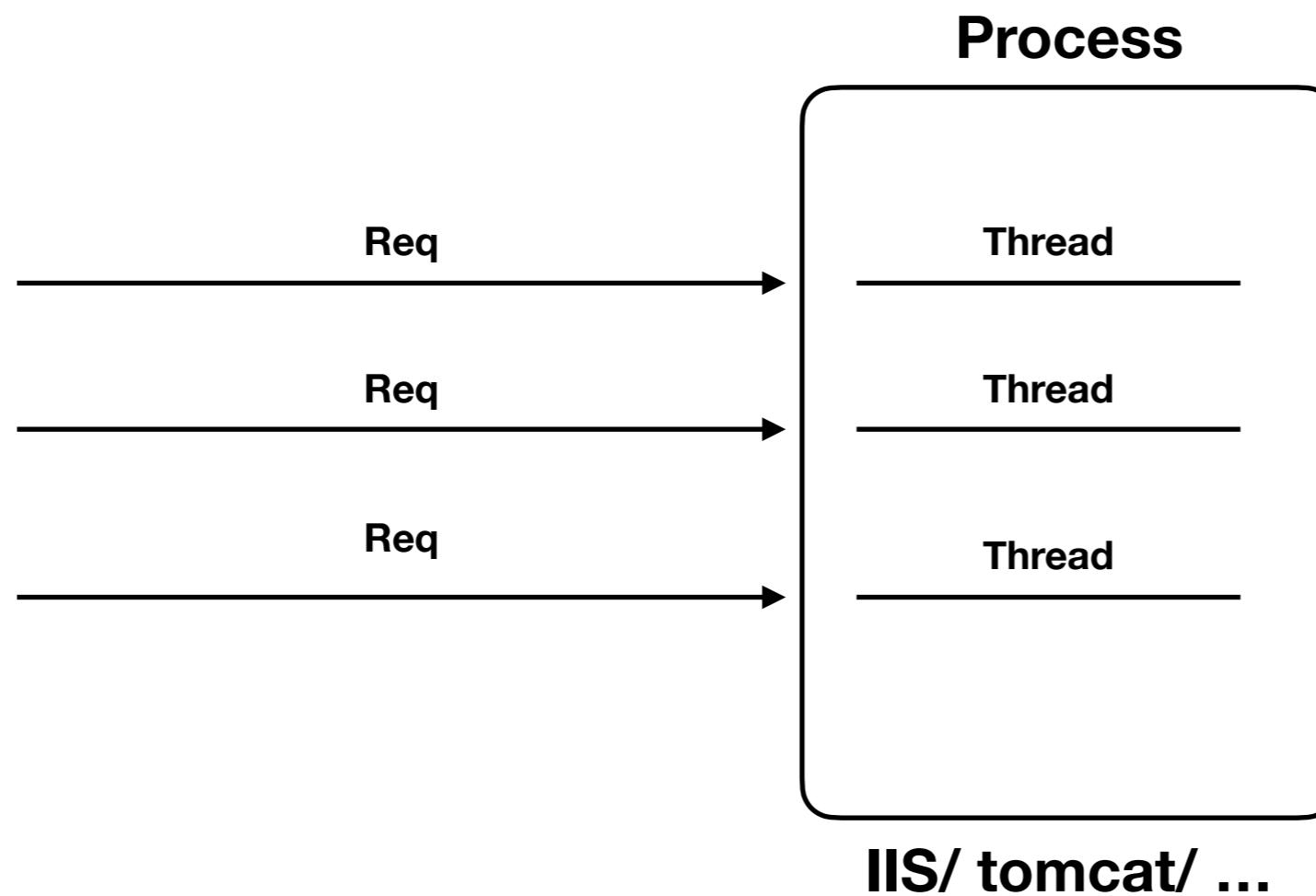


Design 4 - Cache pub-sub

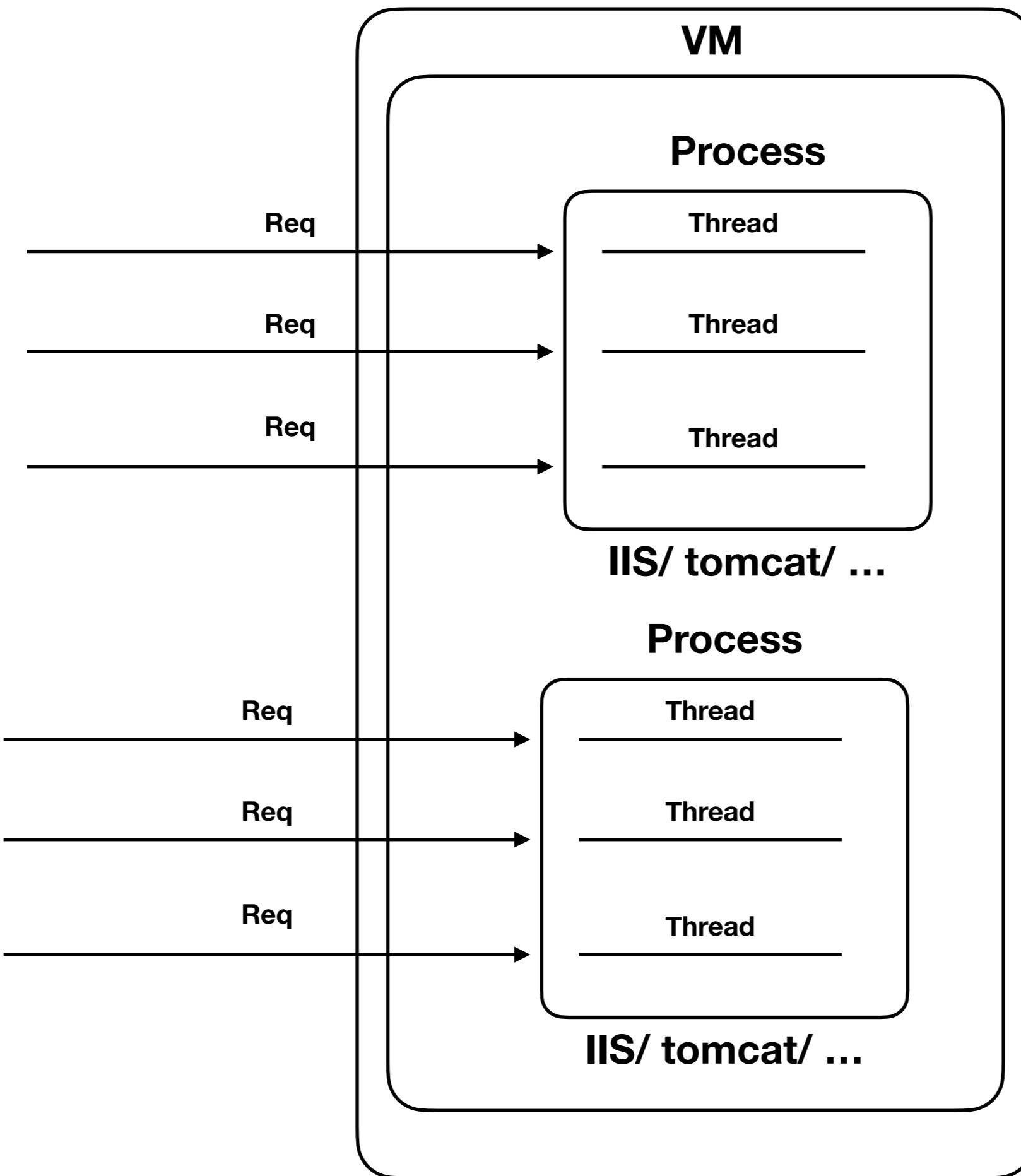


32 bit

- **4 gb process memory**
- **1350 + threads**
- **300 ~ thread**



Web Garden

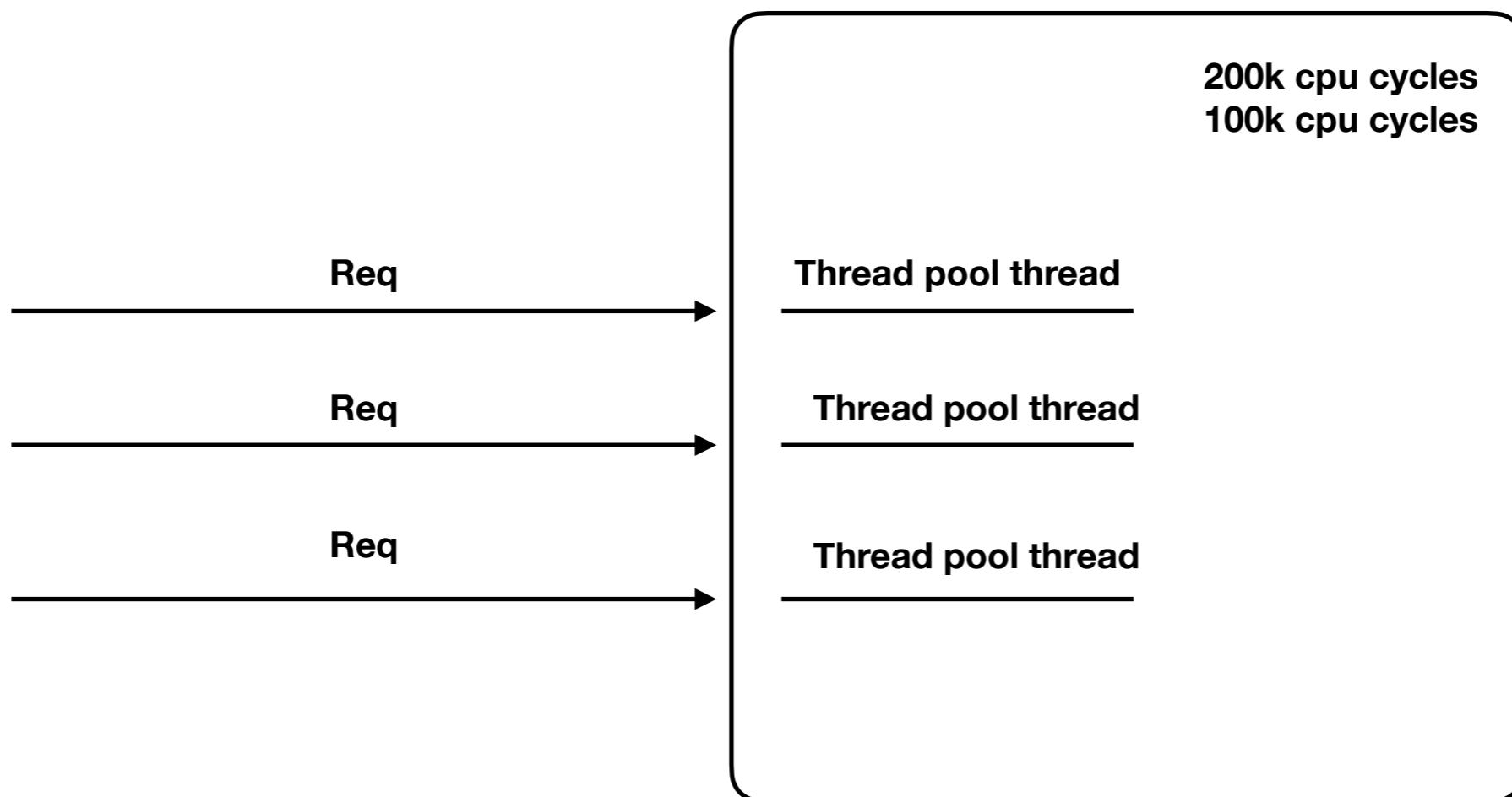


600 ~ request

32 bit

- **4 gb process memory**
- **1350 + threads**
- **300 ~ thread**

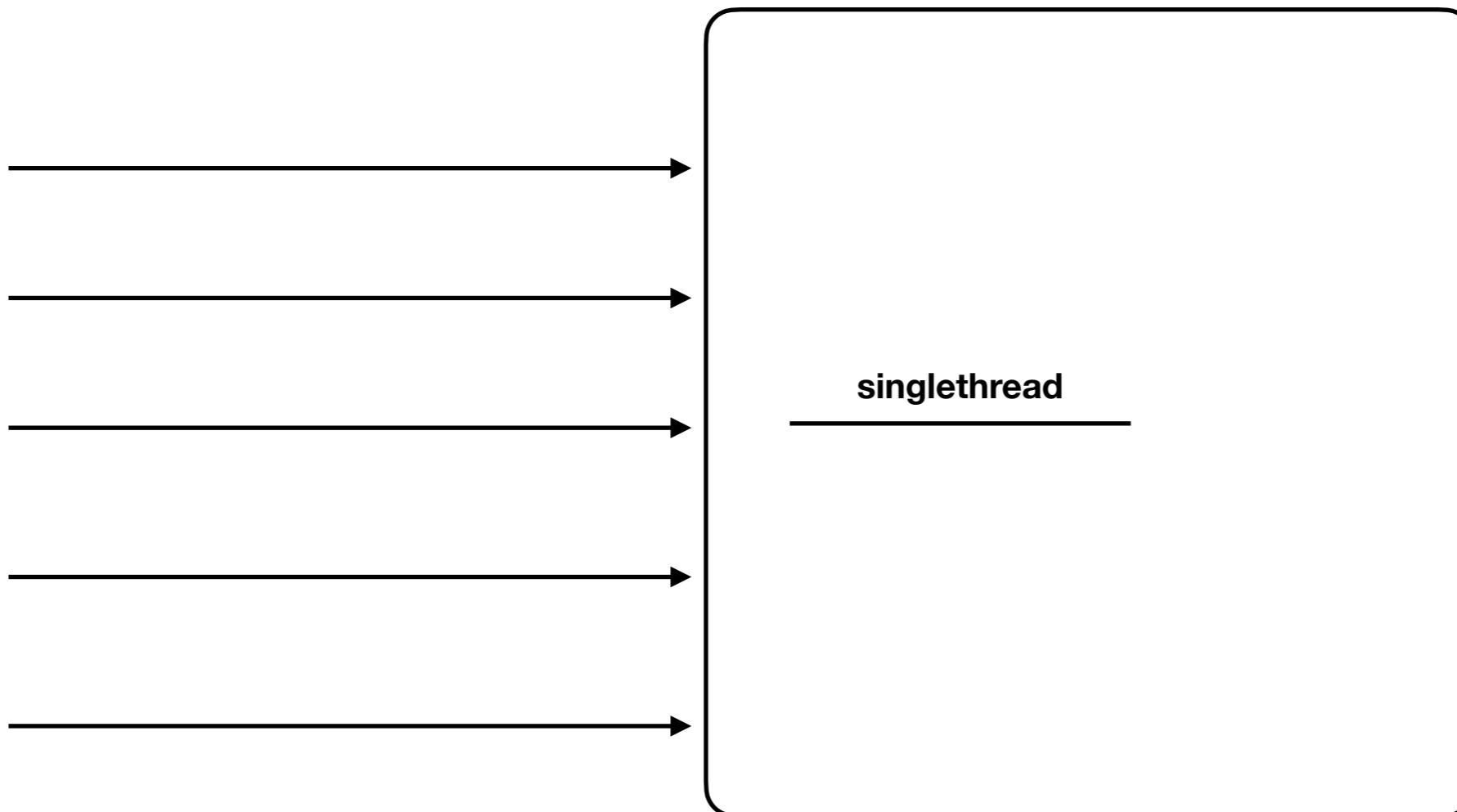
Process



IIS/ tomcat/ ...

+ 50k connections

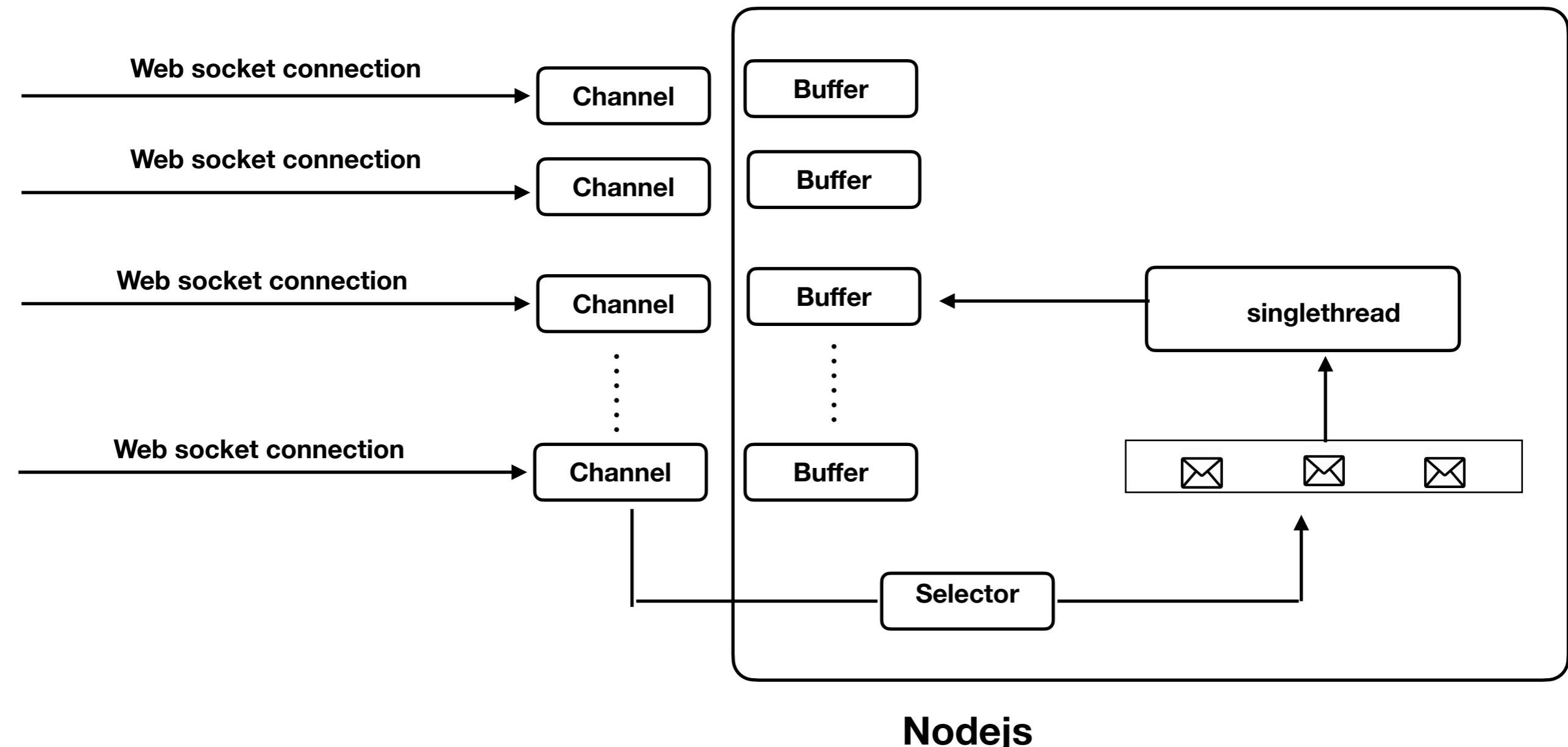
Process



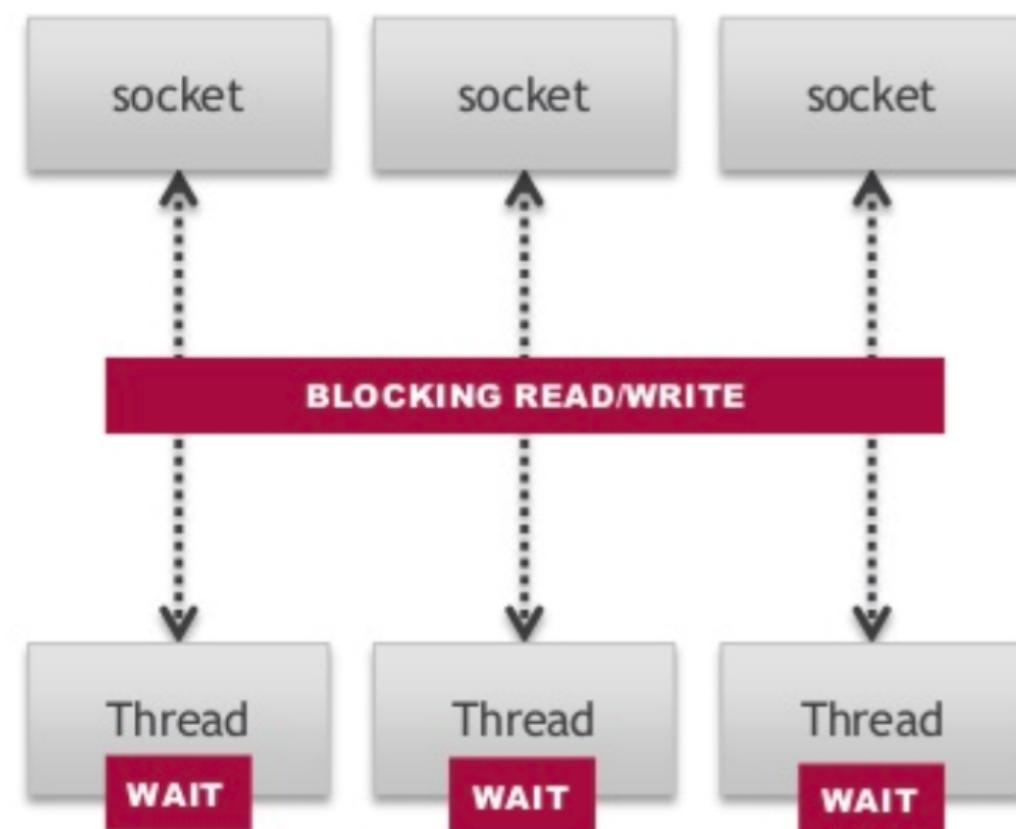
Nodejs

+ 50k connections

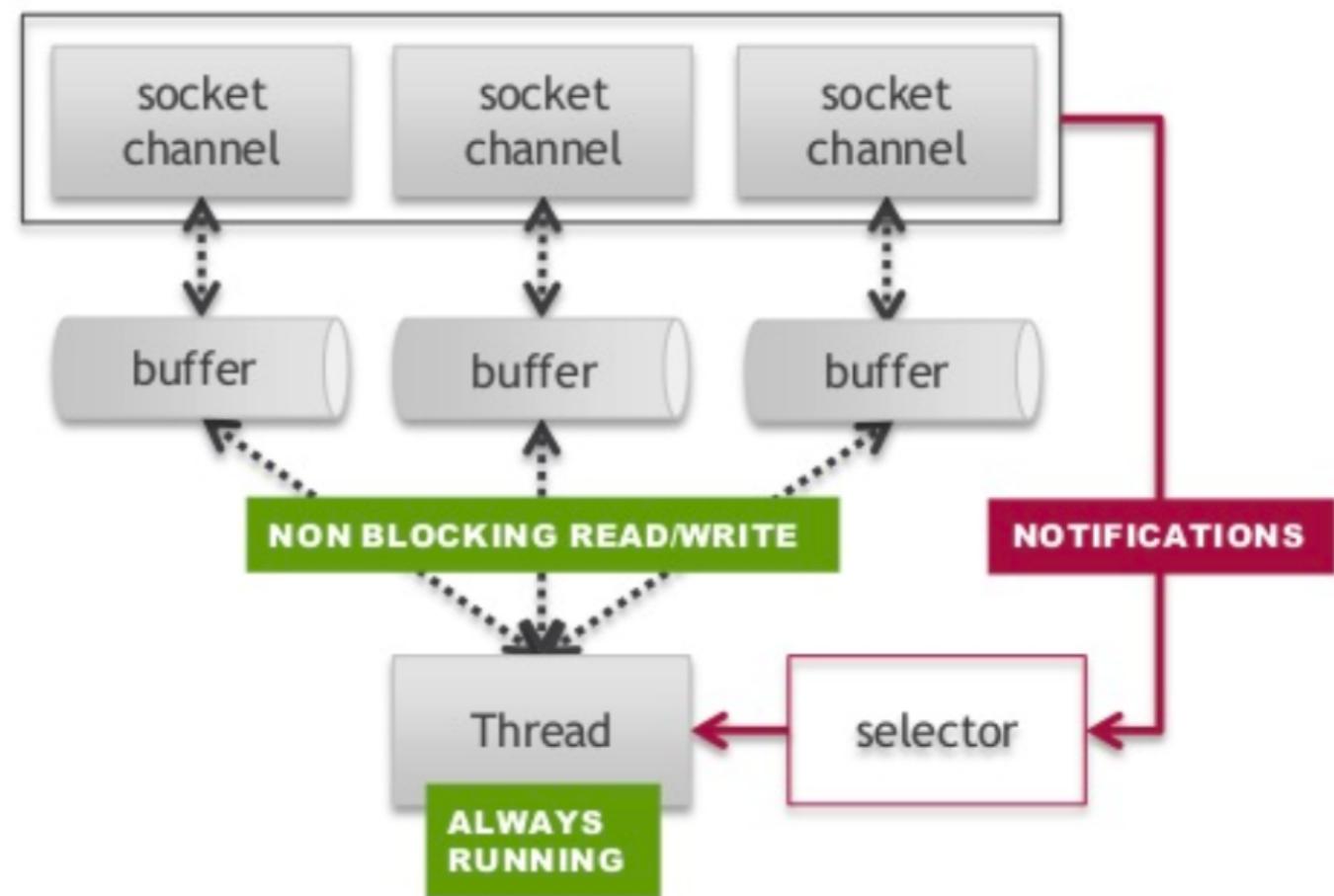
Process



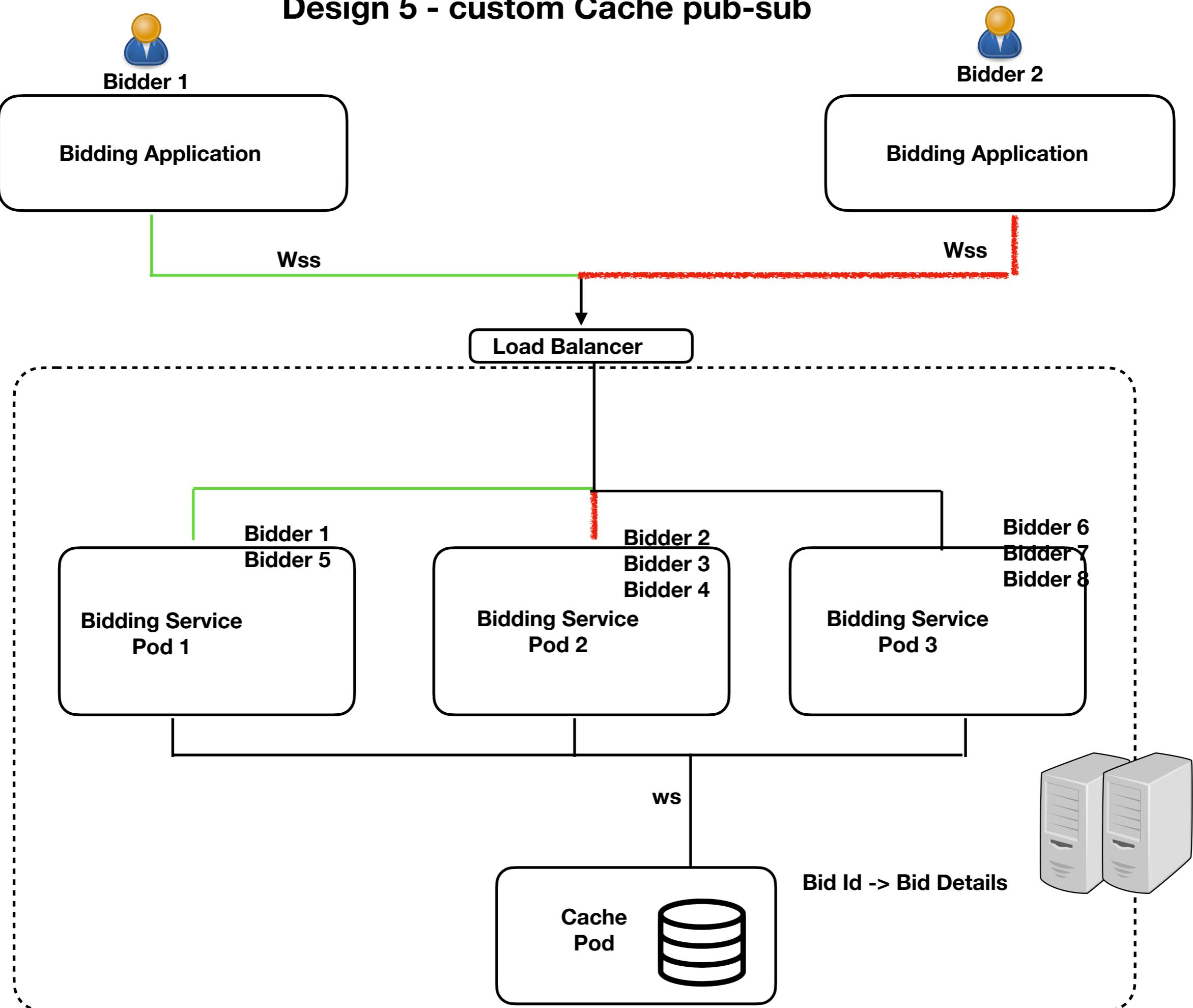
IO (blocking)

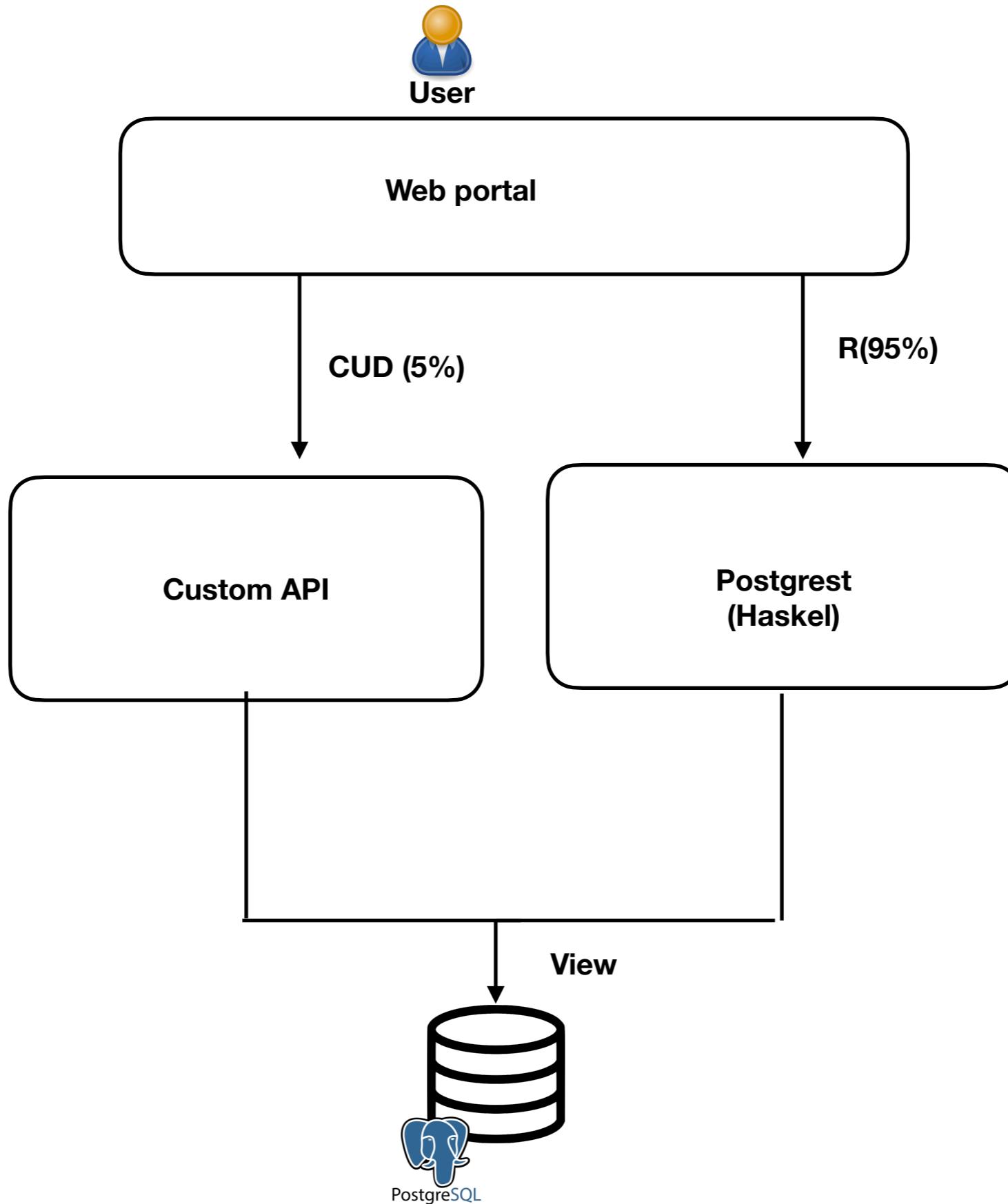


NIO (non-blocking)

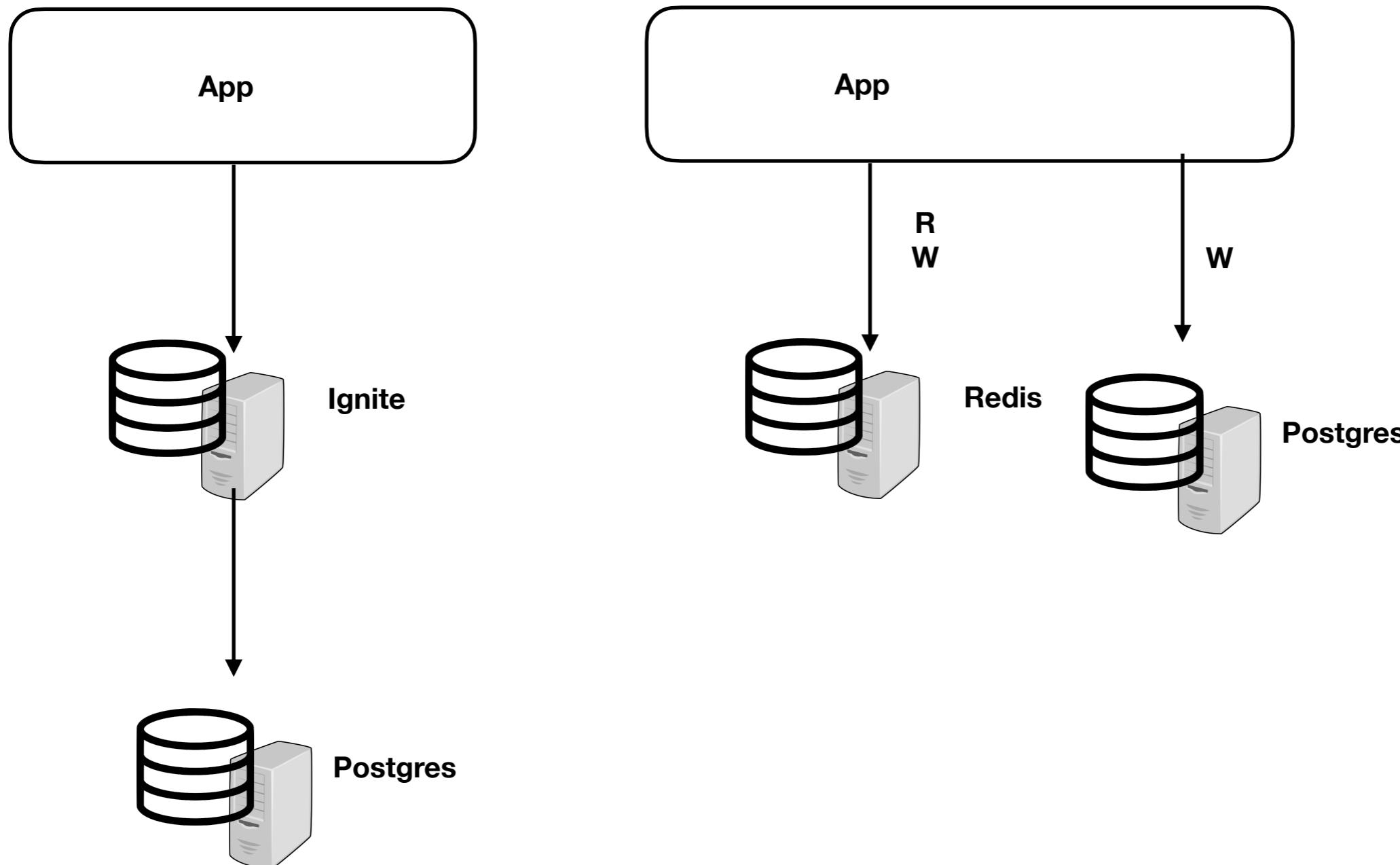


Design 5 - custom Cache pub-sub

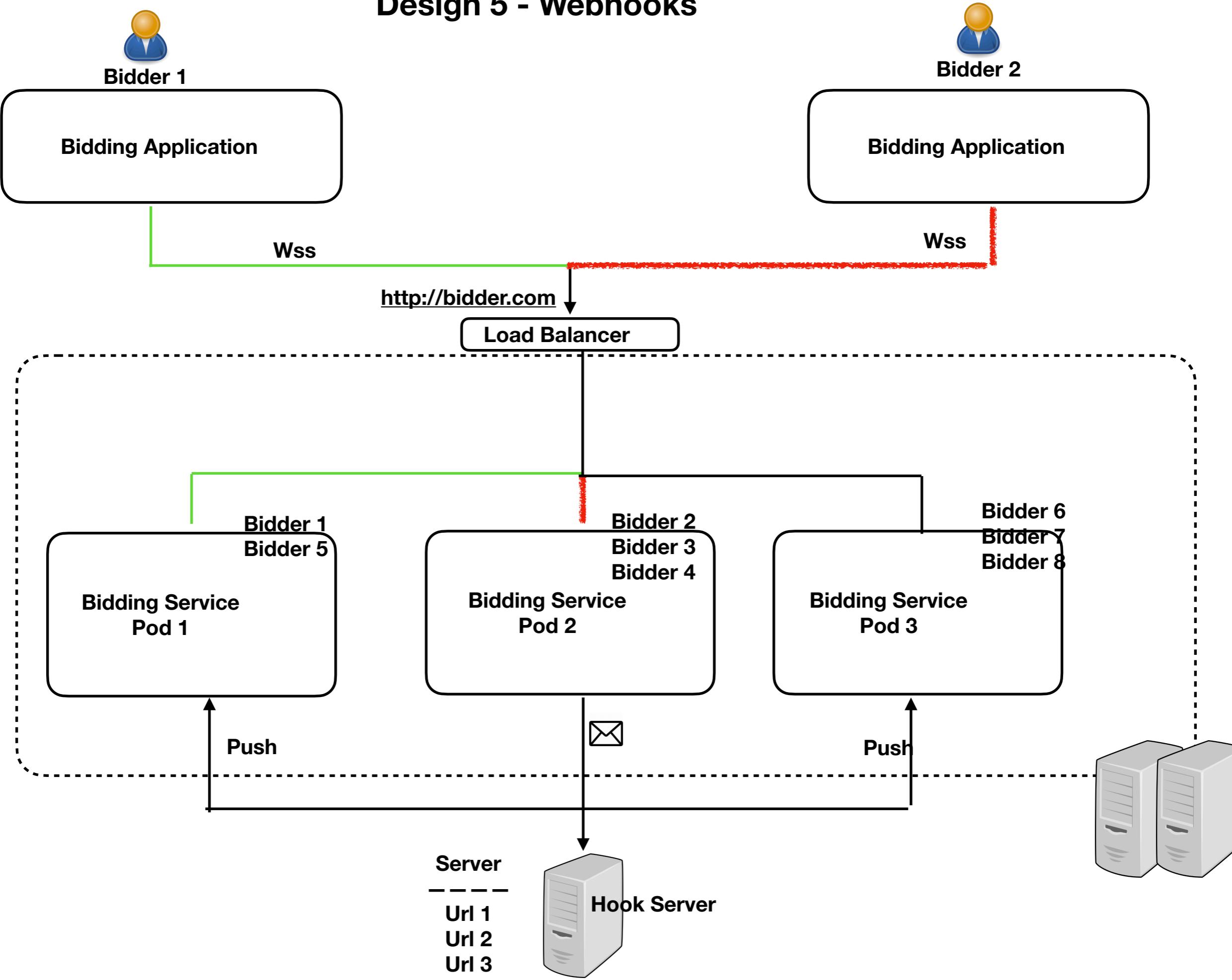




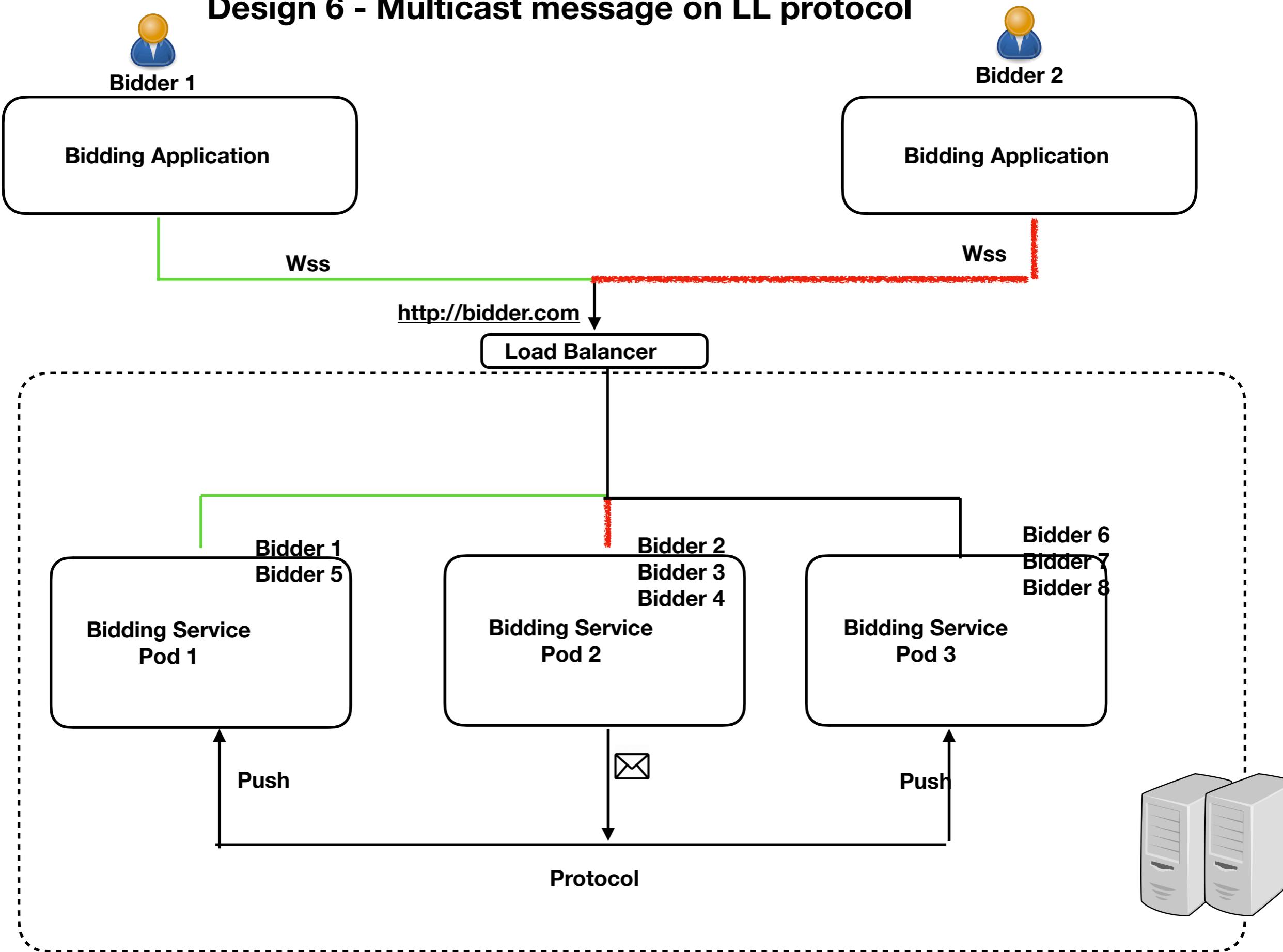
Write through cache



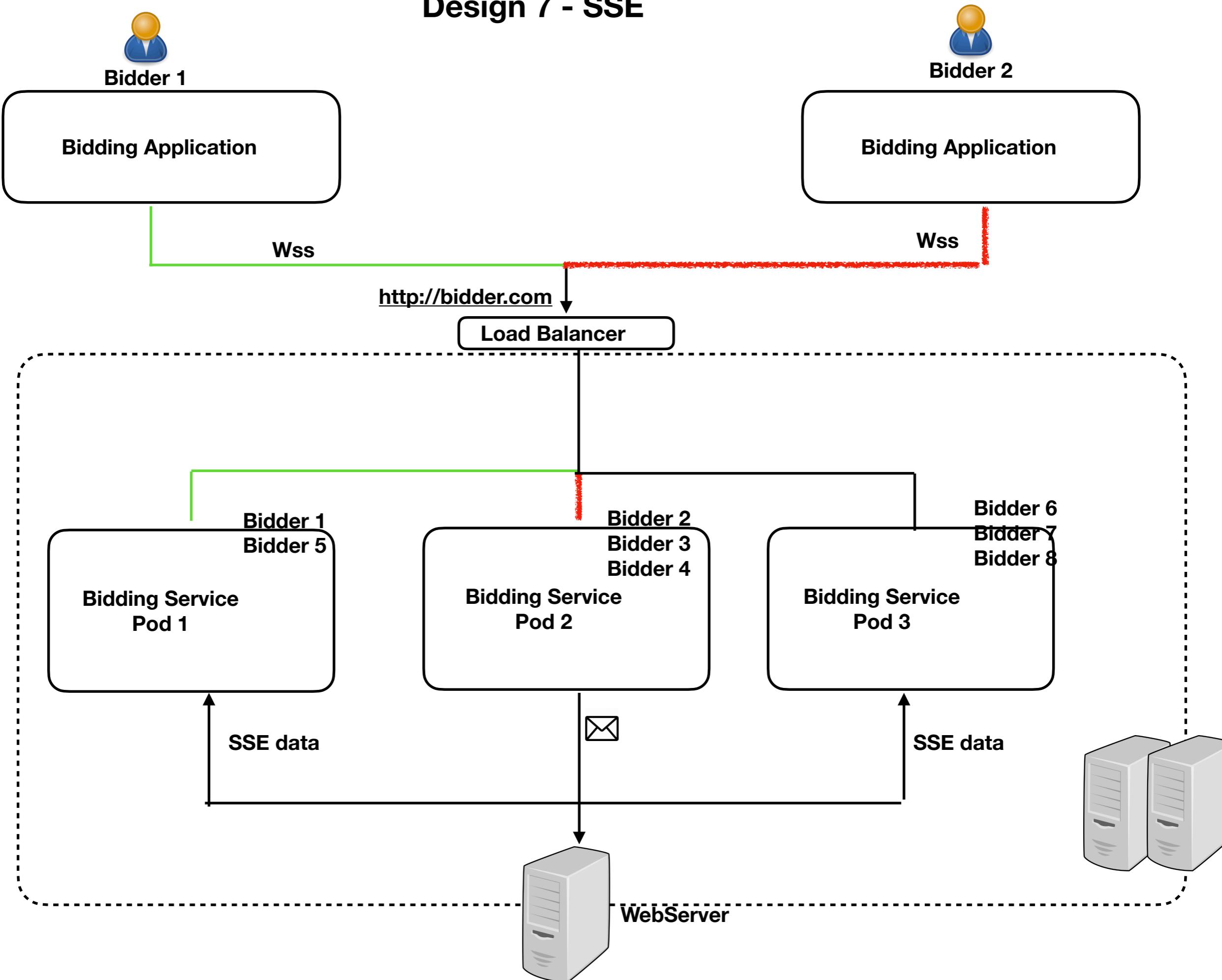
Design 5 - Webhooks



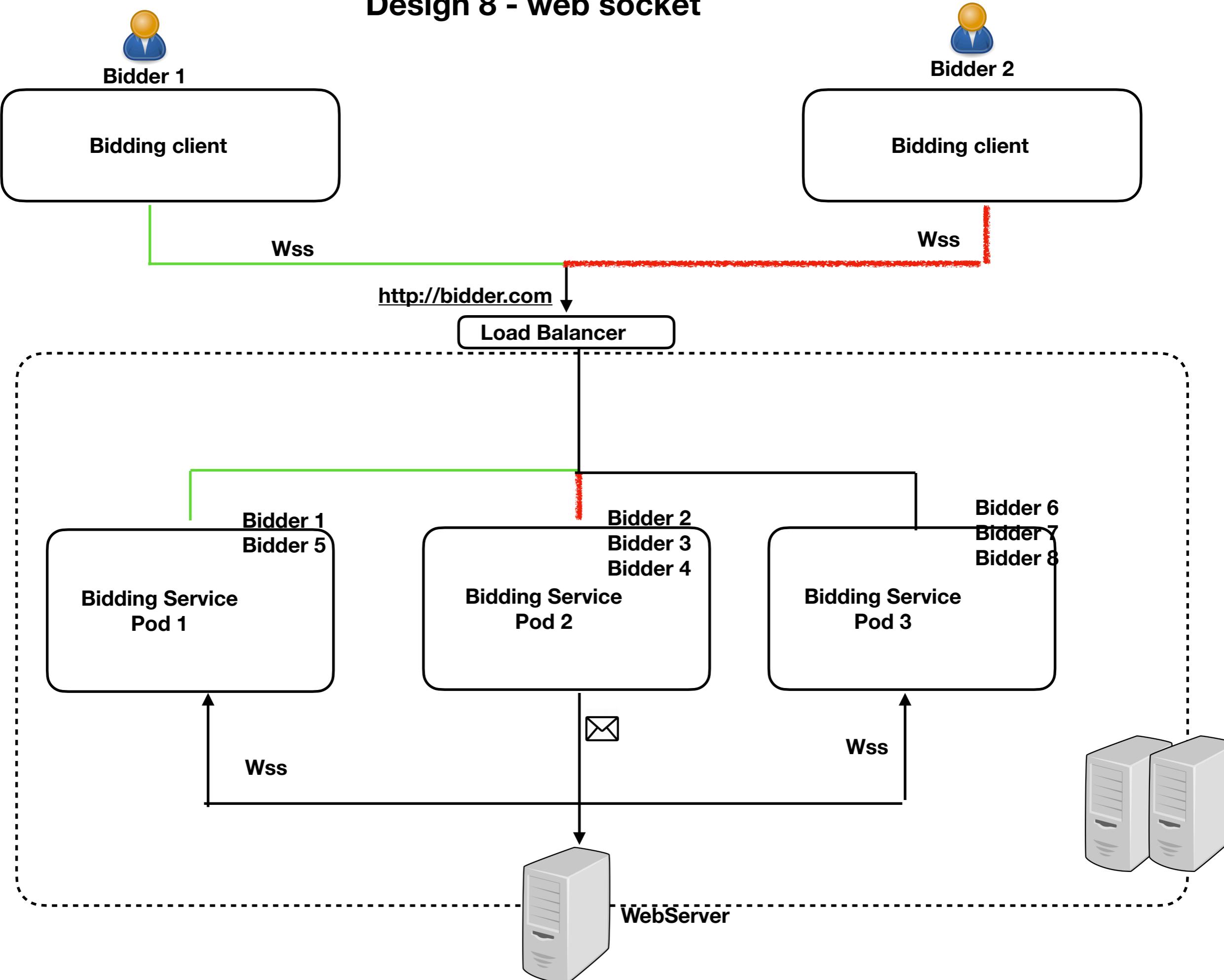
Design 6 - Multicast message on LL protocol



Design 7 - SSE



Design 8 - web socket



Security View



Bidder

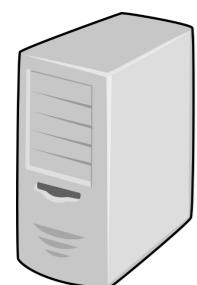
Bidding Application

aws

Cloud

DMZ Zone

Key Vault



Application Zone

Domain Event Bus

Database Zone



Wss

L4 - Firewall

Load Balancer

L7 - WAF

Reverse Proxy
(throttling)

FastAPI

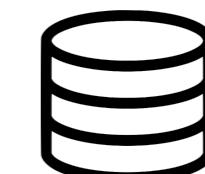
Pydantic

Bidding Service

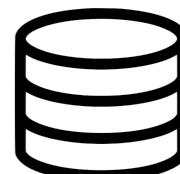


kubernetes

Security
Token
Service

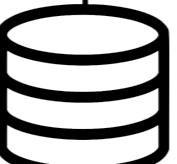
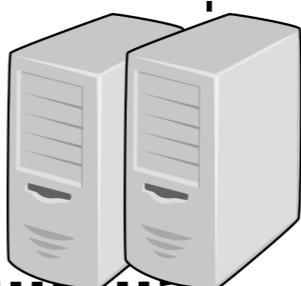


Credential
Store

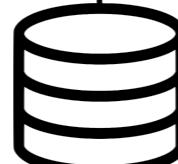


Claim
Mapping

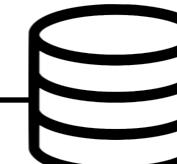
user->permission



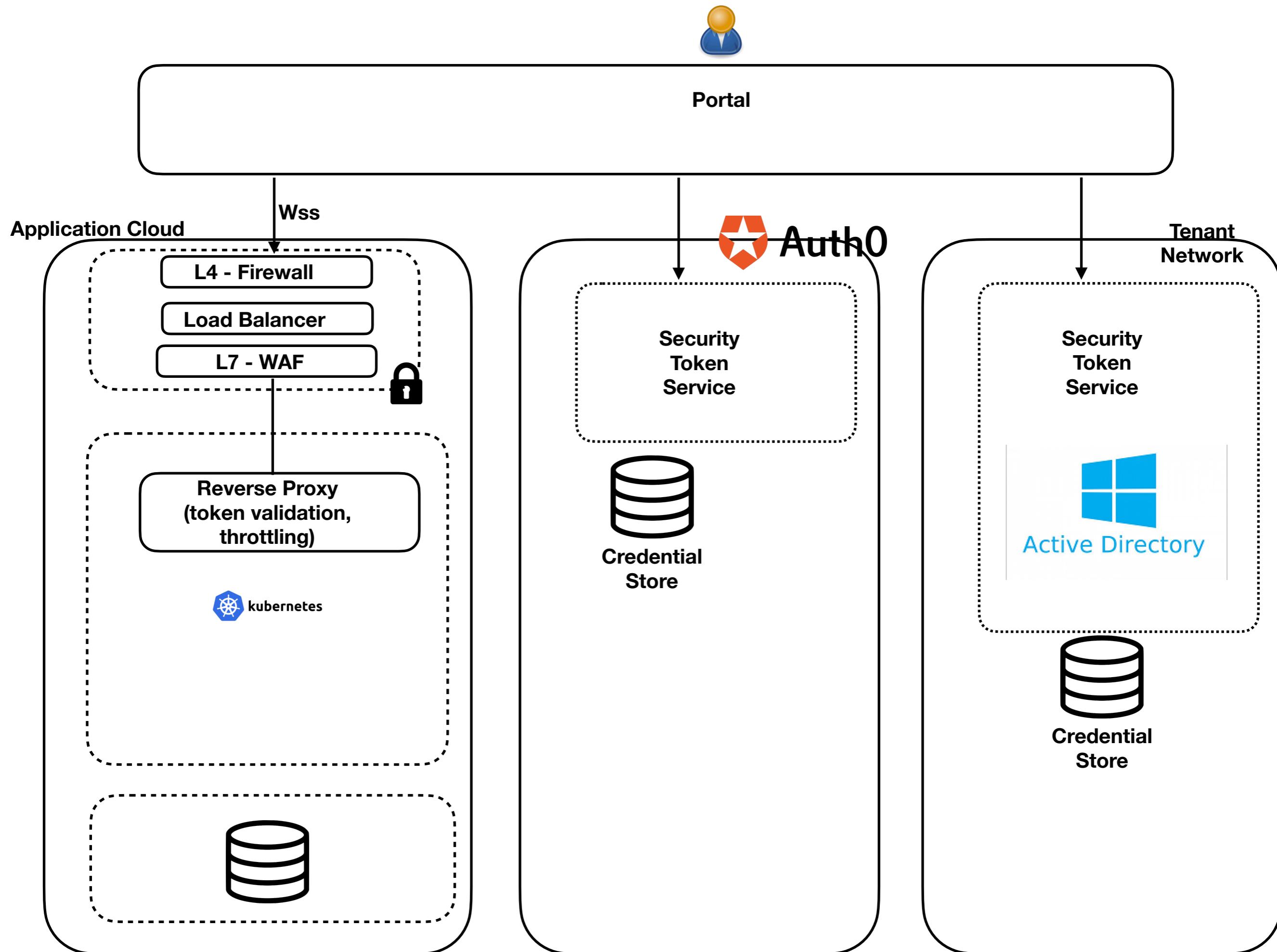
Cache
Service



Rdbms
Store



Replicated
Server



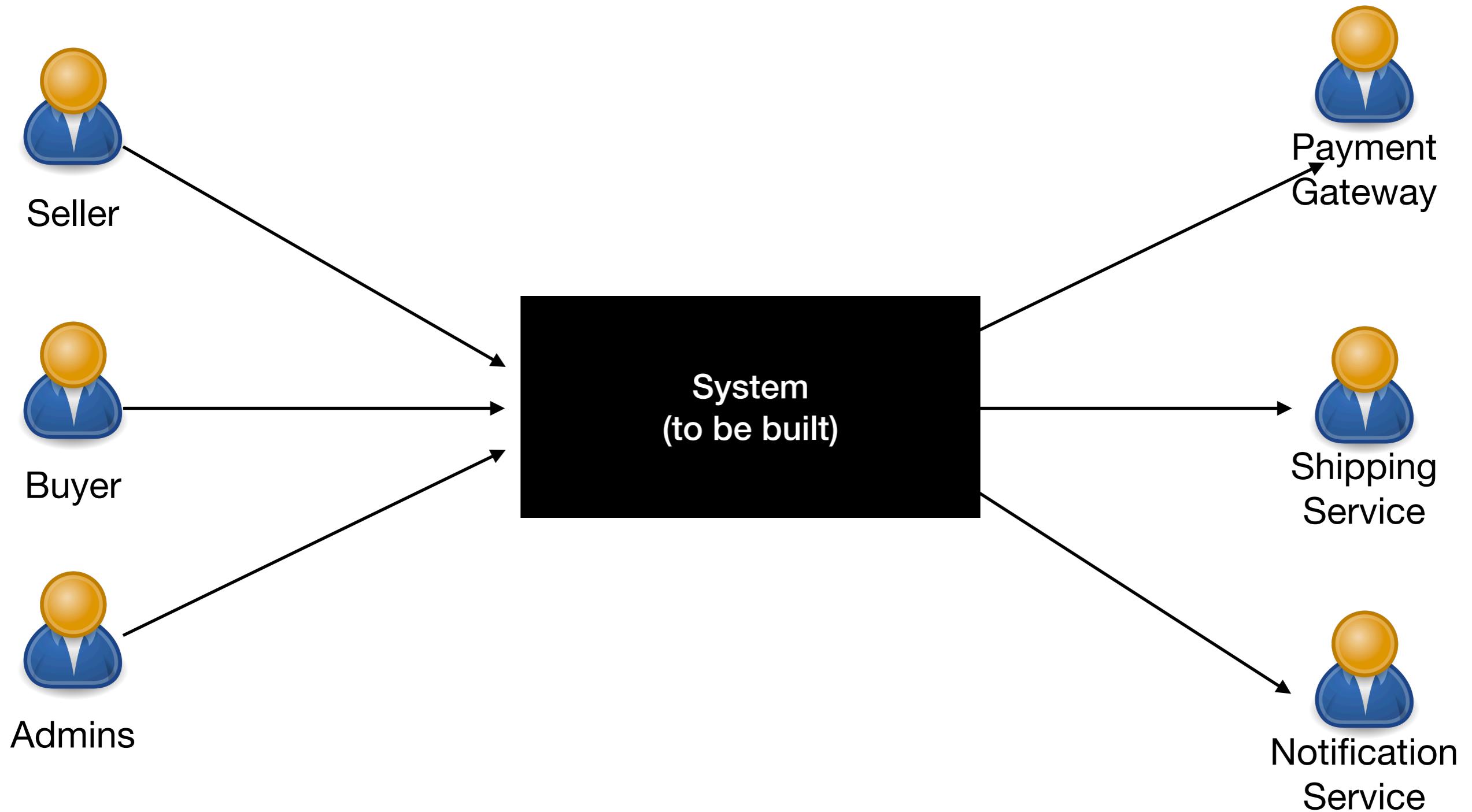
- Context view (black box)
- Logical View (white box)
- Security View (

Ecommerce Case Study

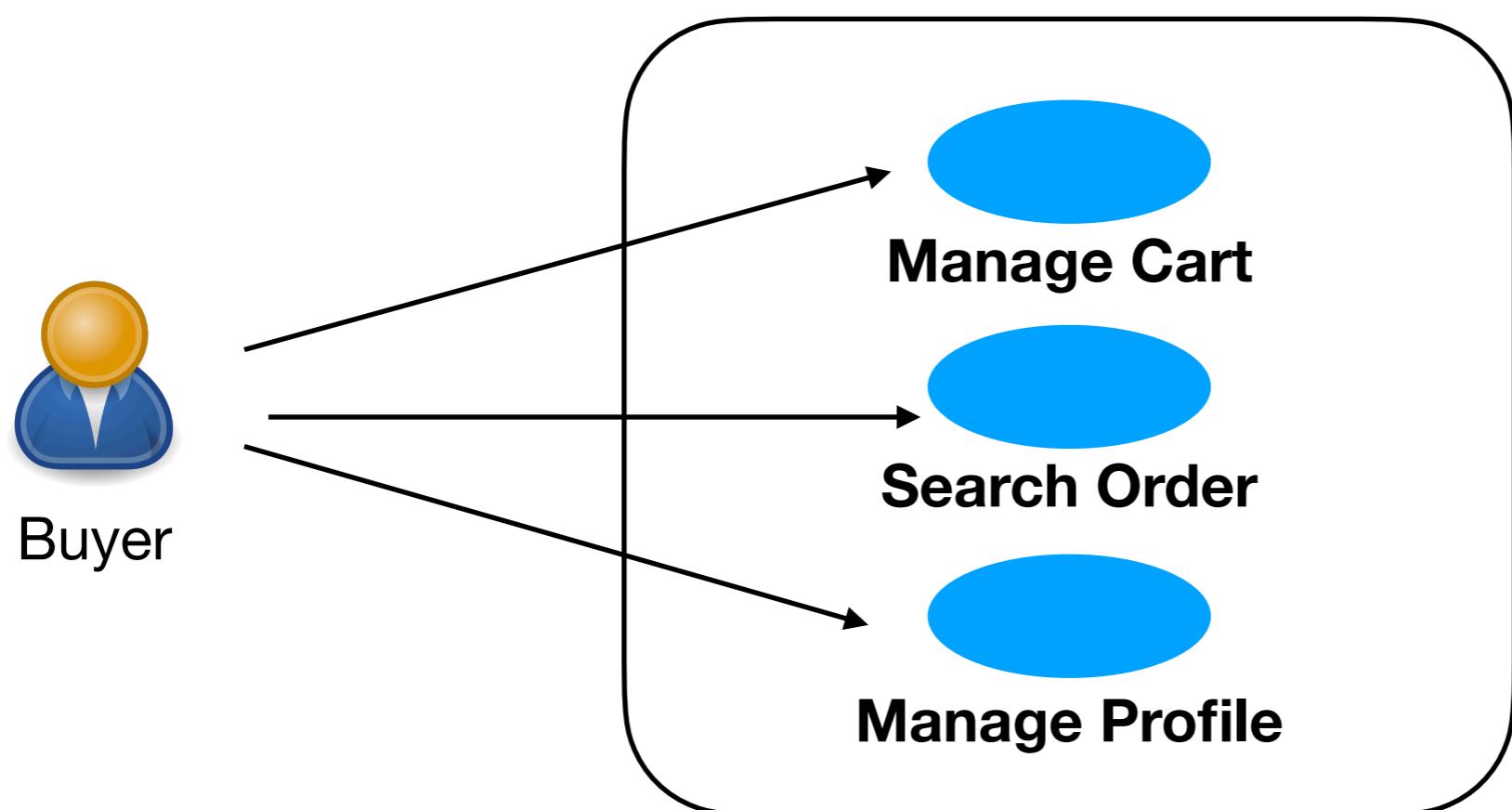
Architectural Requirements

- # Context View
- # Functional View
- # Constraints (removed)
- # Quality View
- # Assumptions (removed)

Context View



Functional view



Quality View

Quality	Source (who)	Stimulus (action)	Artifact (which)	Environment (context)	Response (output)	Measure (scale)
Scale	As a Buyer	I should be able to place an order	In the portal	During peak load.	The order is placed.	Should support 1 million users

Architectural Design

- # Logical View
- # Deployment View
- # Security View (removed)

Logical View

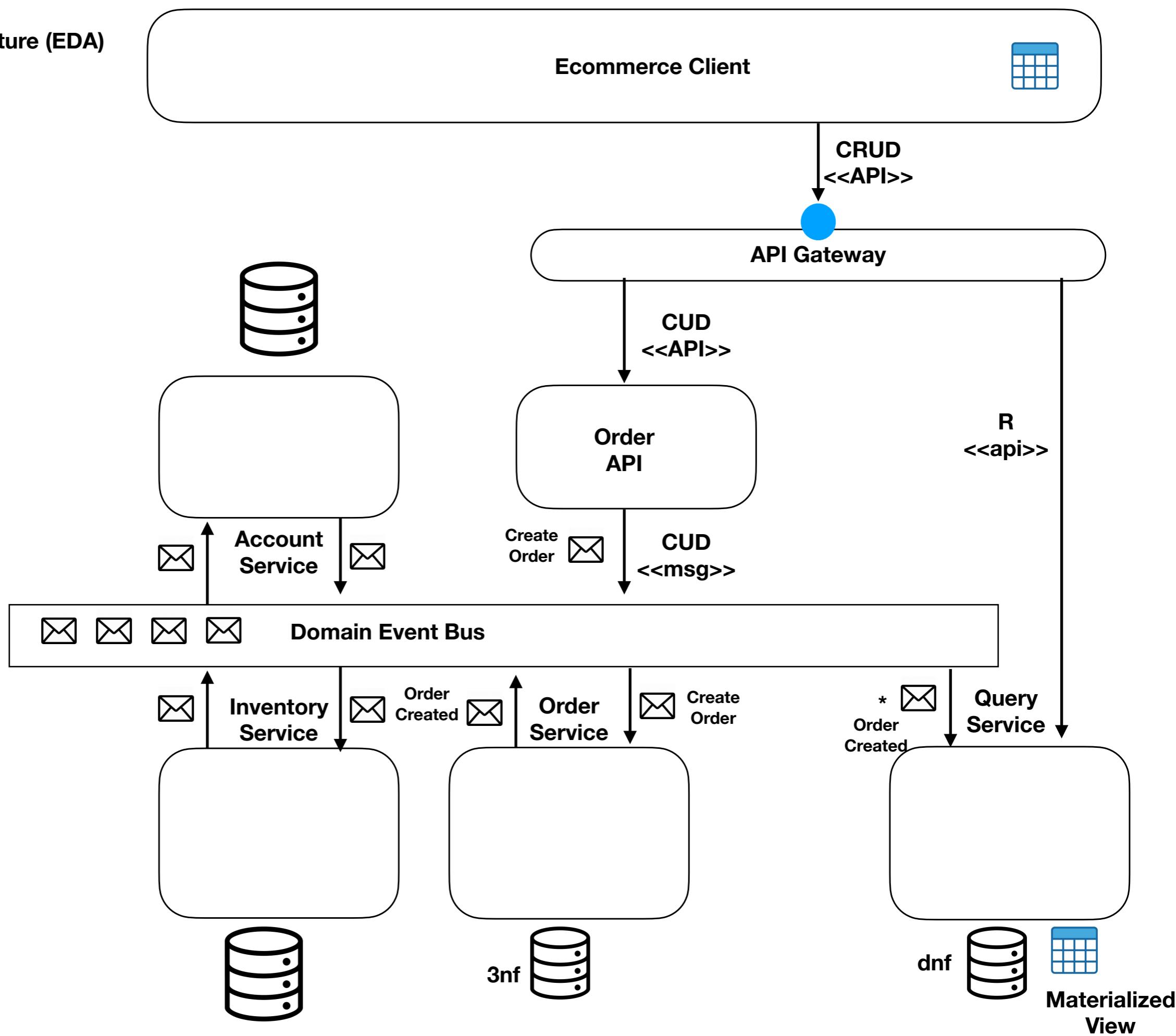
- # System Decomposition**
- # Persistence approach**
- # Compute approach**
- # Communication approach**
- # cross cutting approach**

Event driven Architecture (EDA)

- Domain Events
- Event Sourcing
- Event Storming

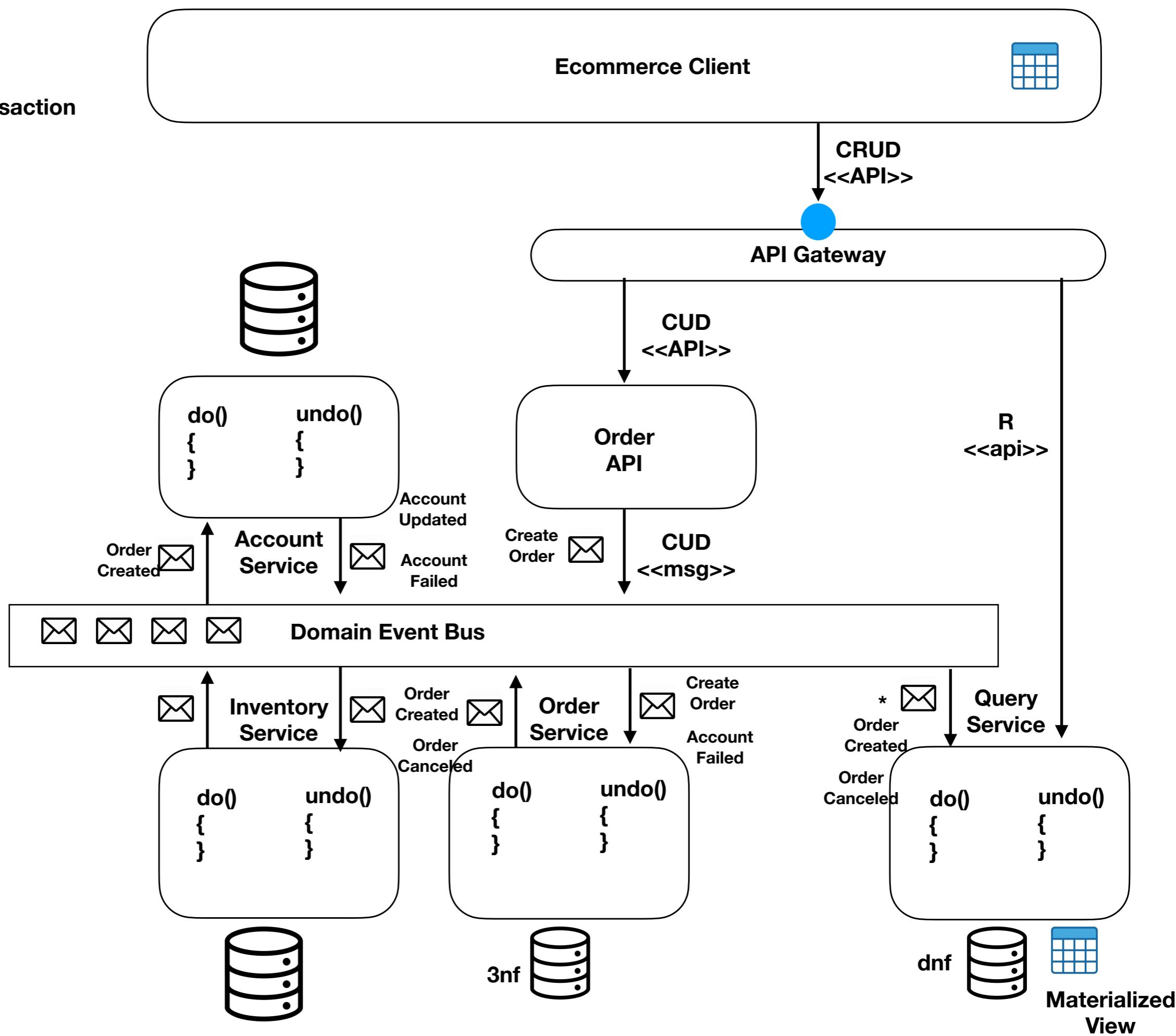
DDD

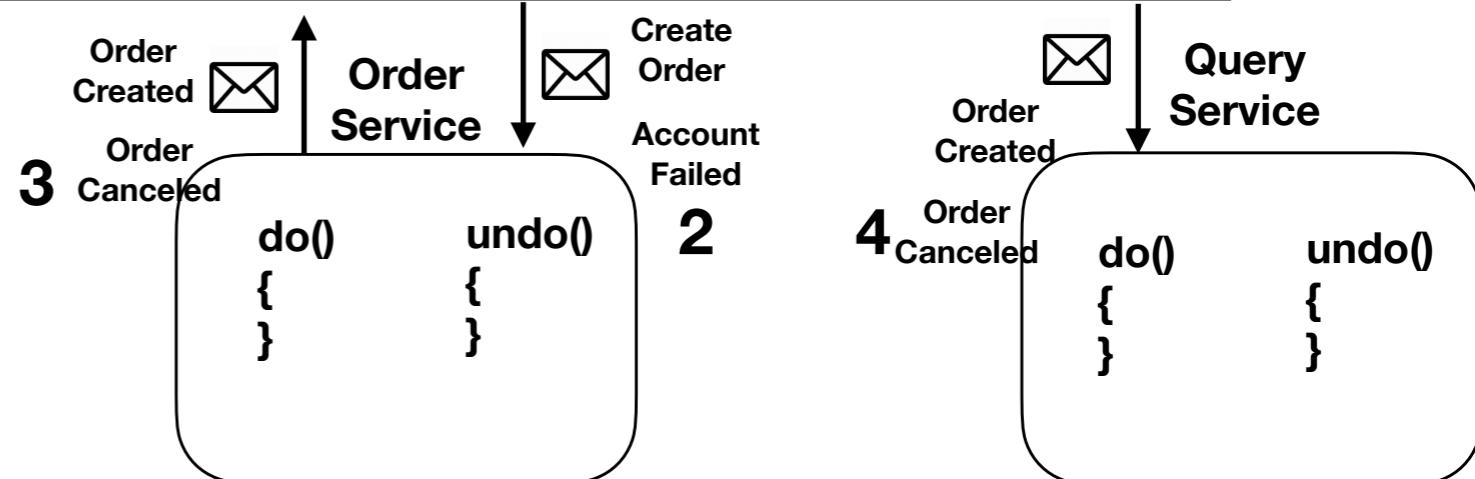
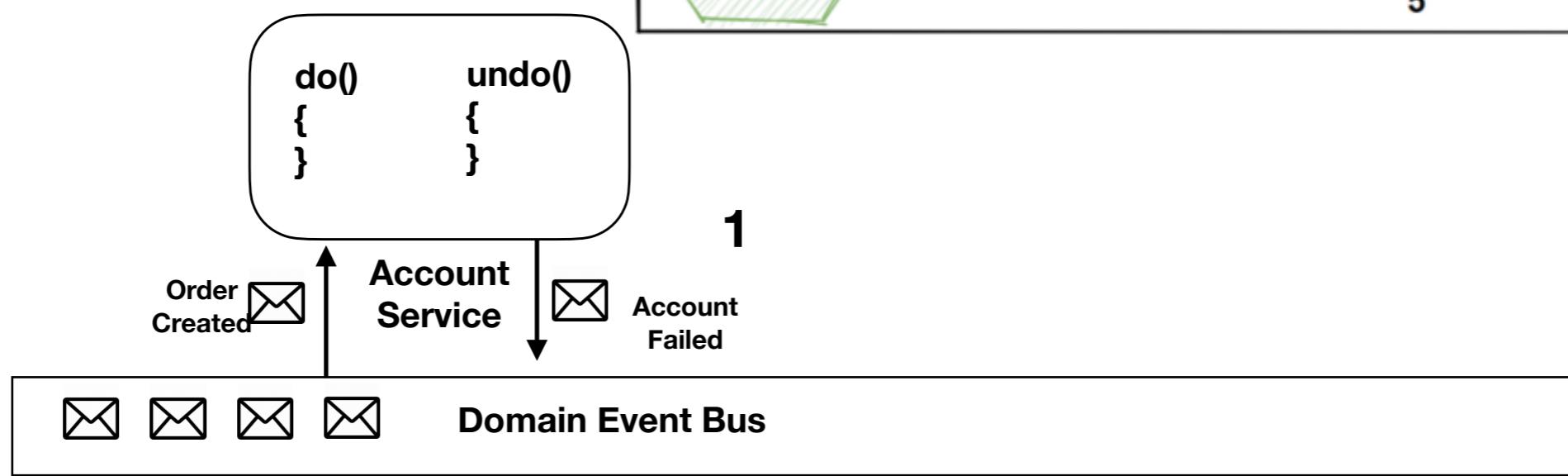
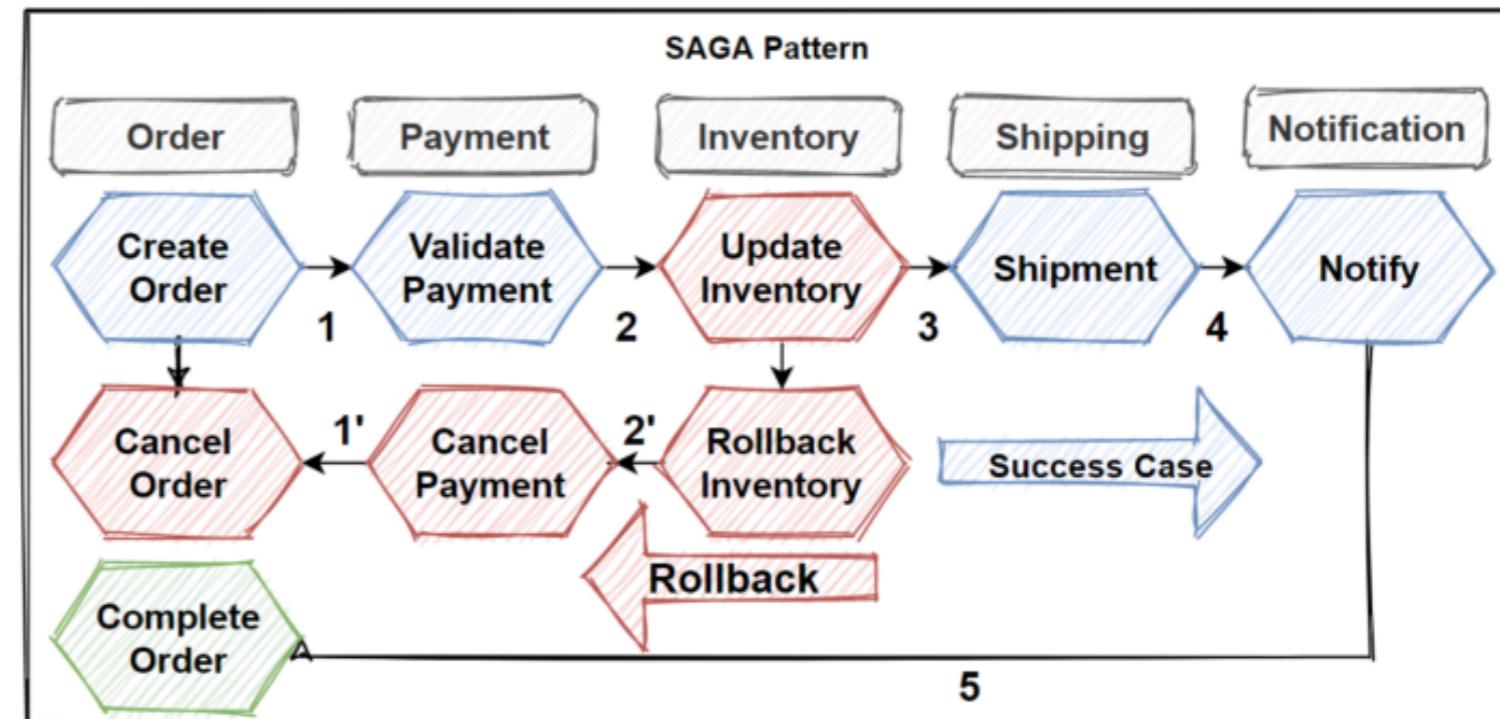
- ubiquitous lang
- Domain Events

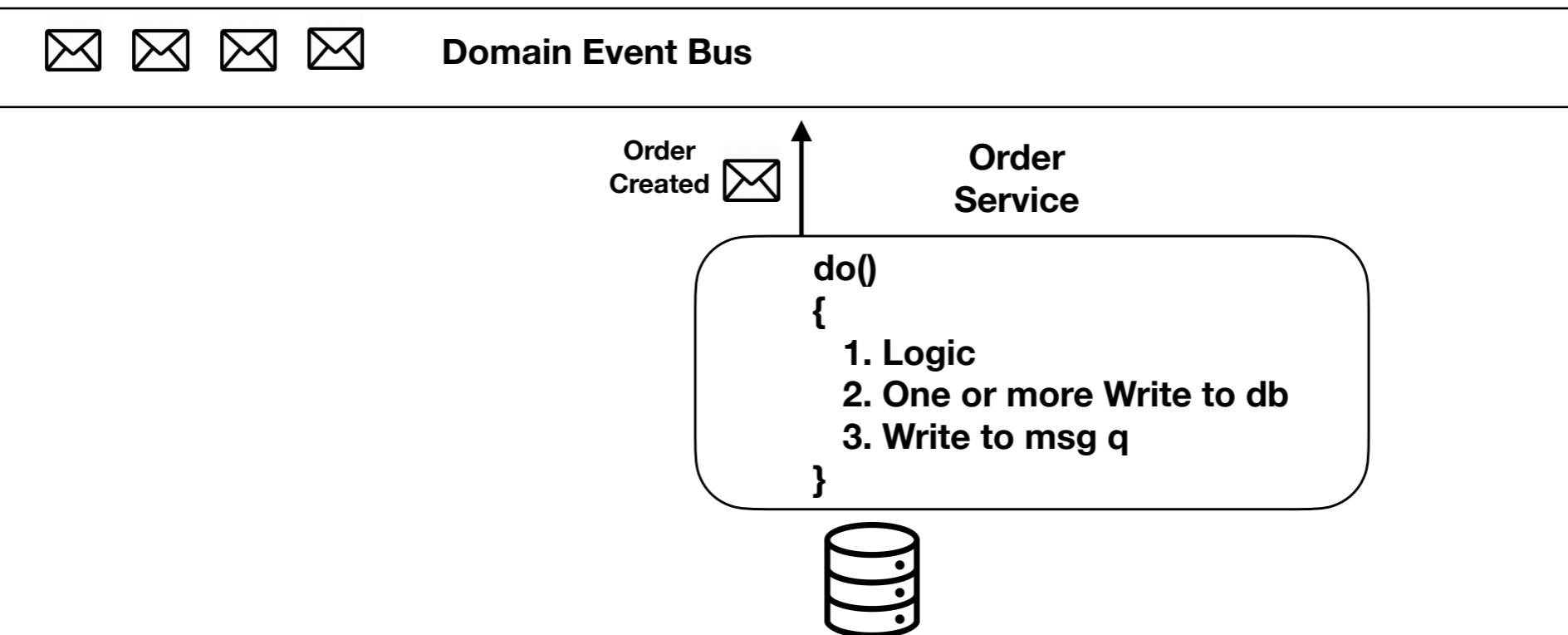


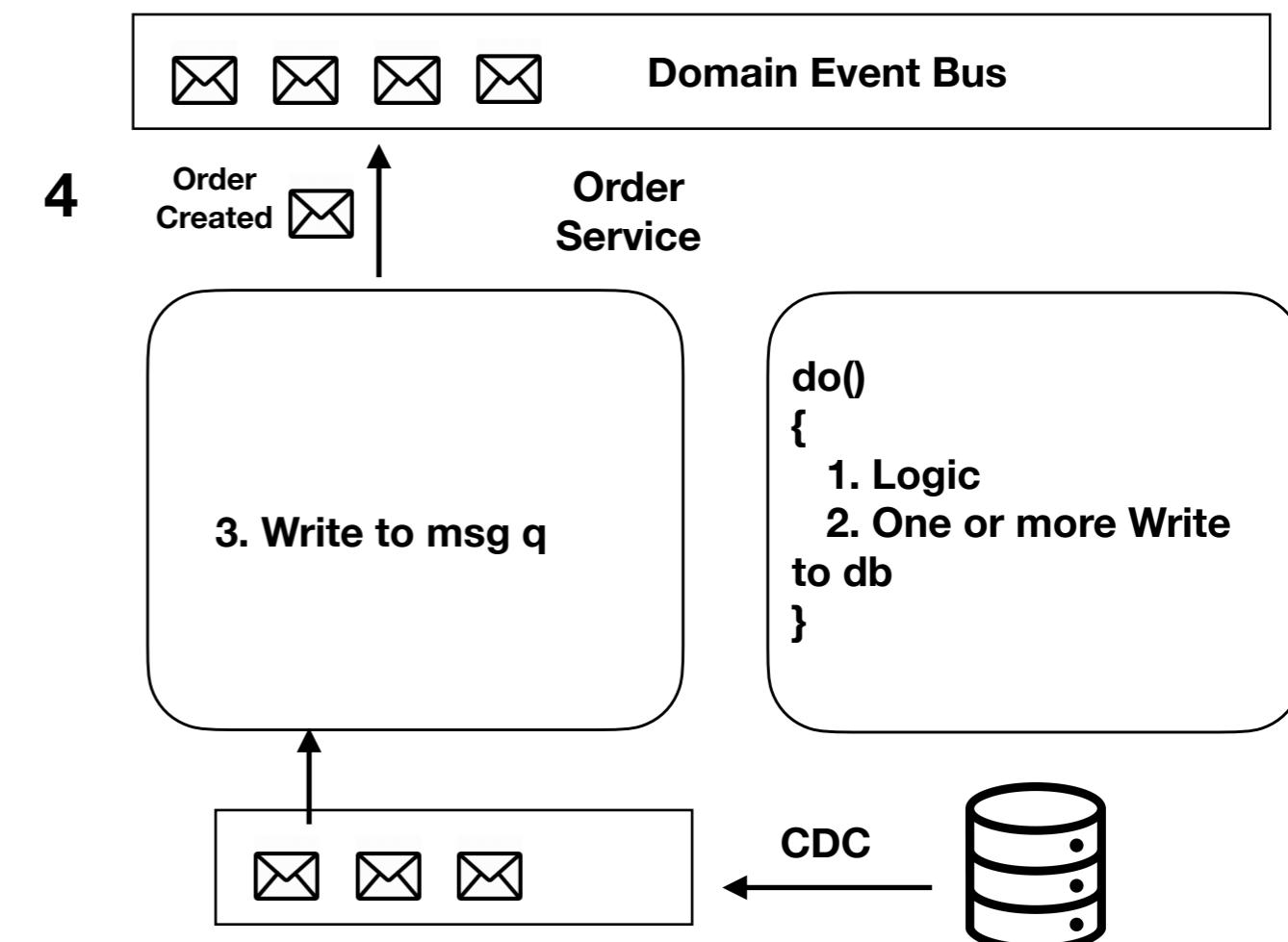
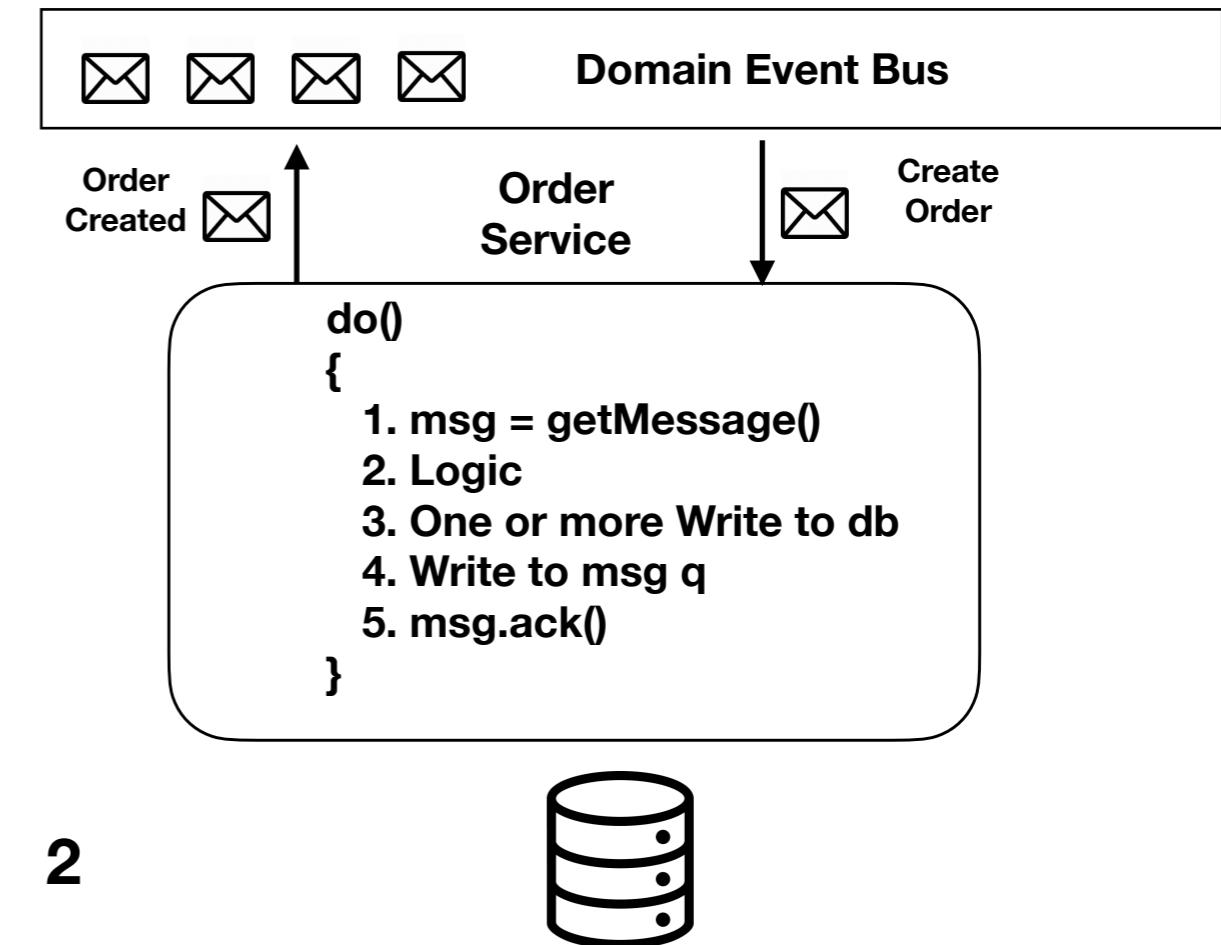
Transactions

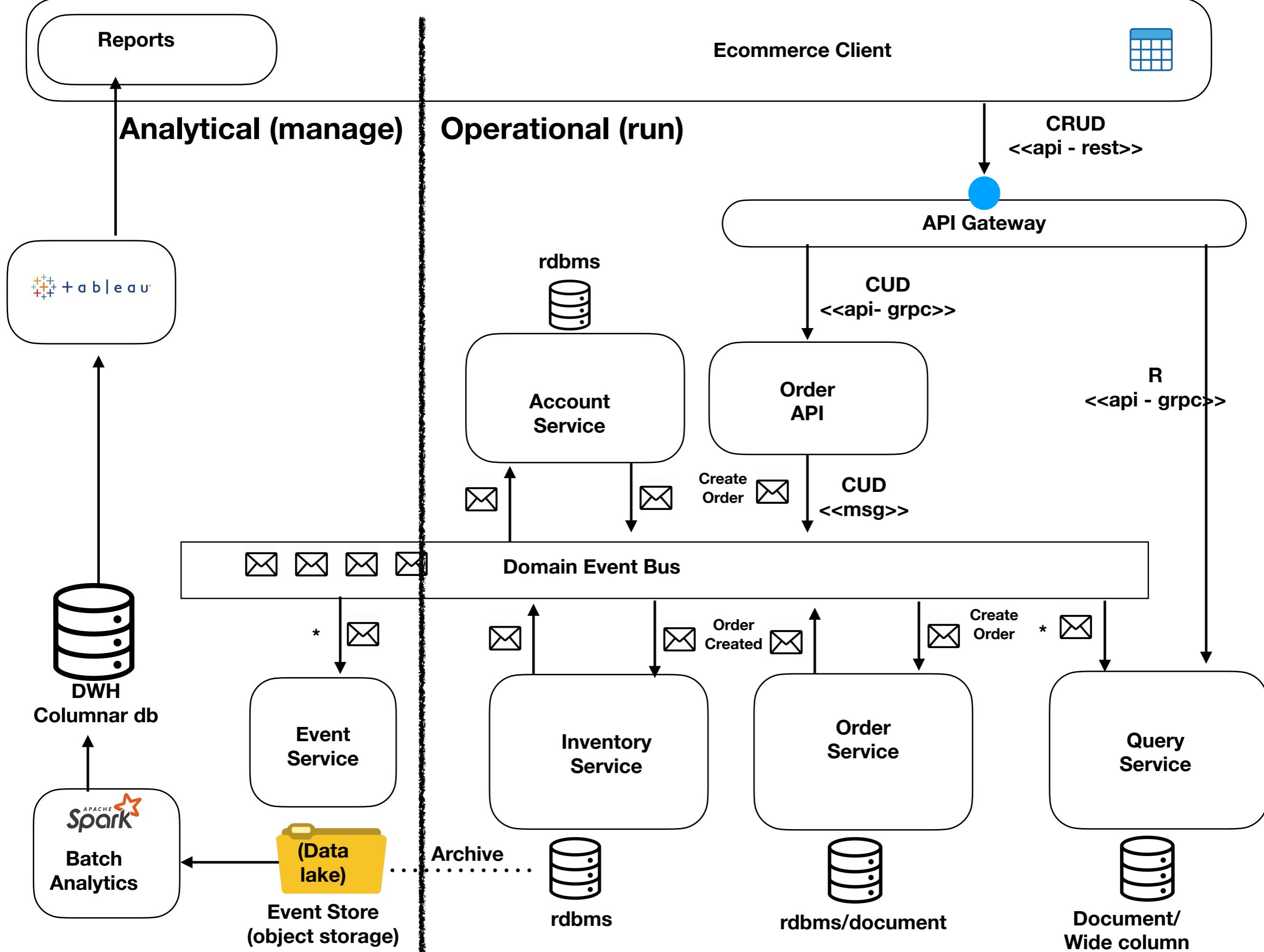
- 1 phase commit
- 2 phase commit
- compensatable transaction
- SAGA

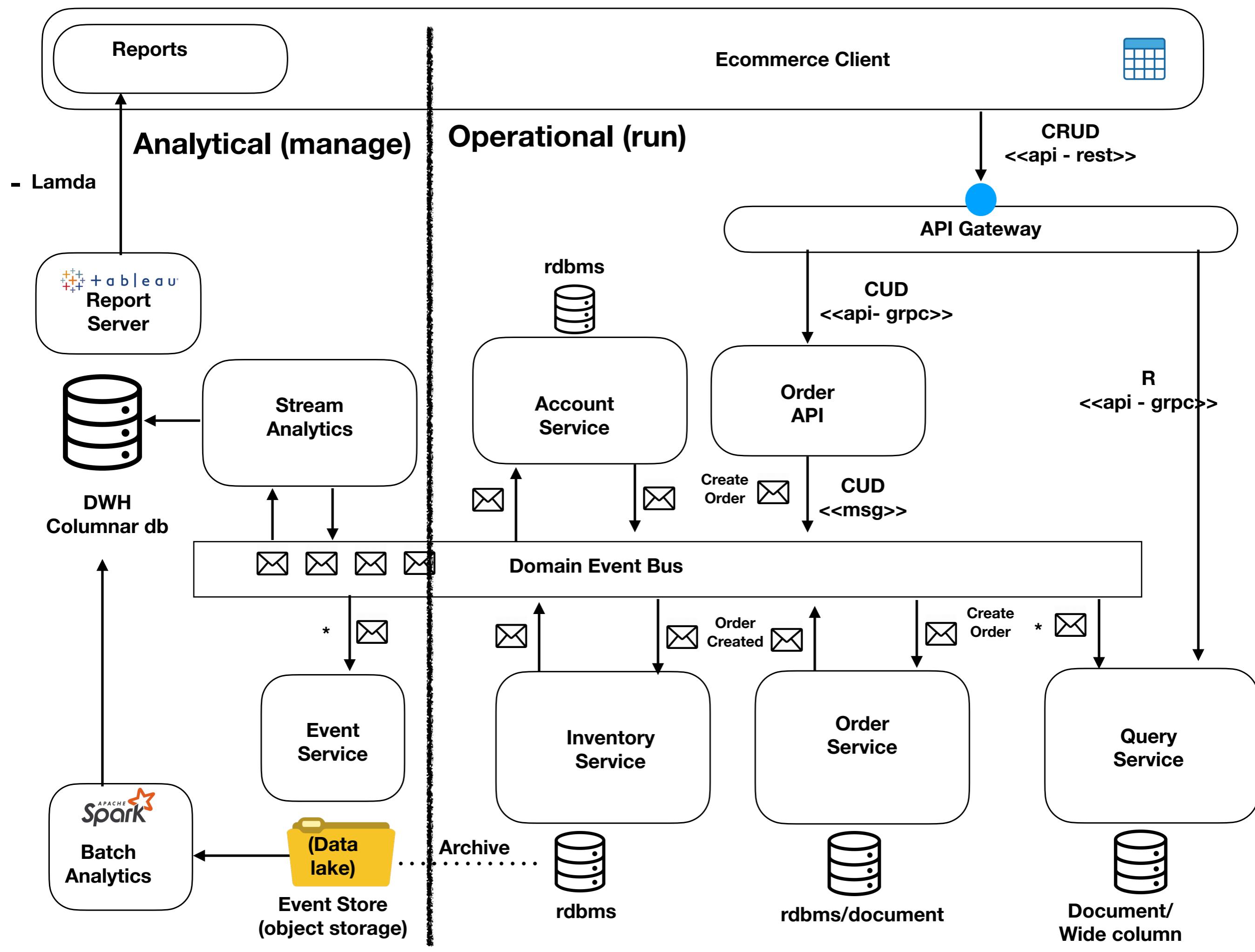


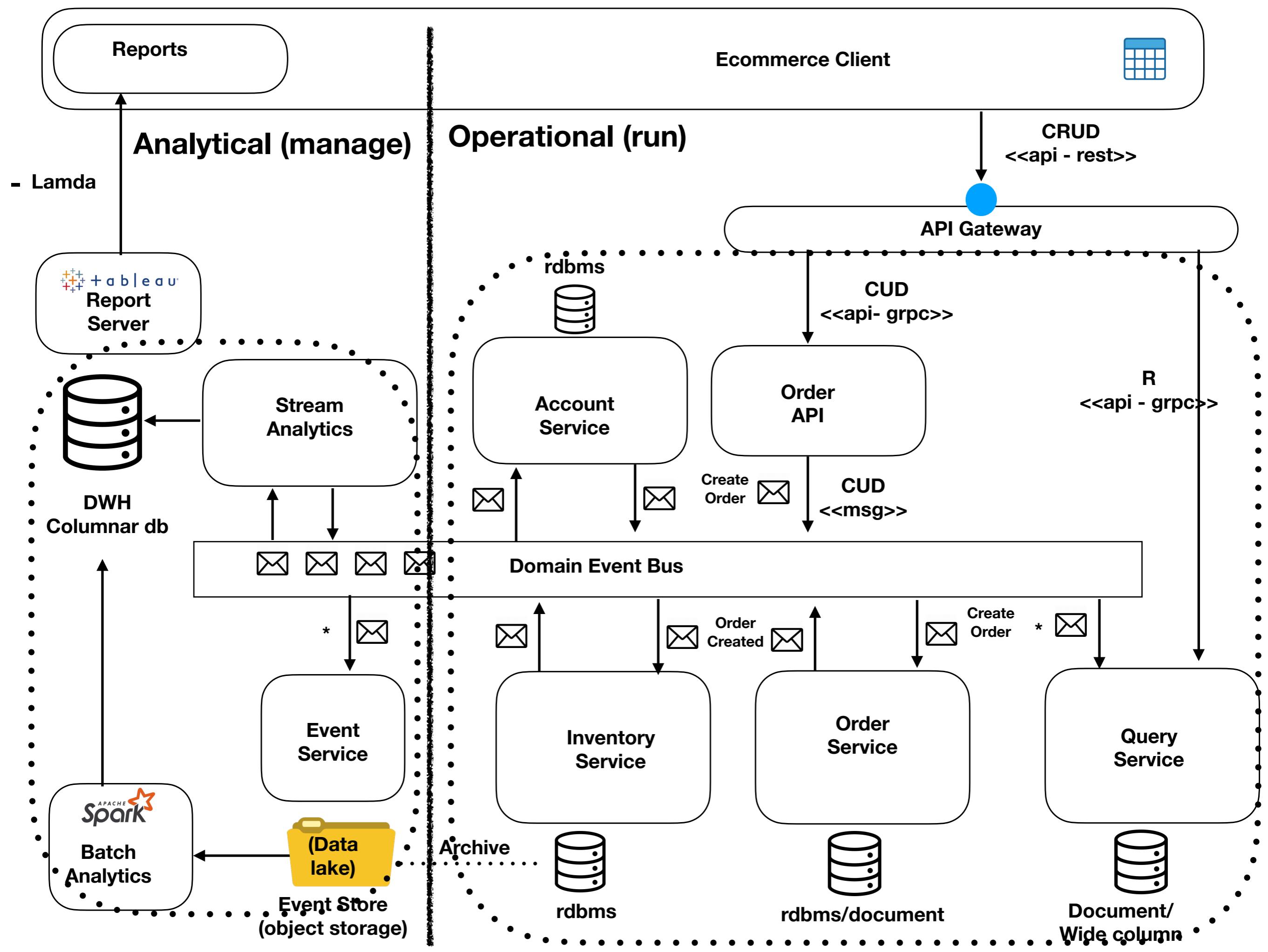






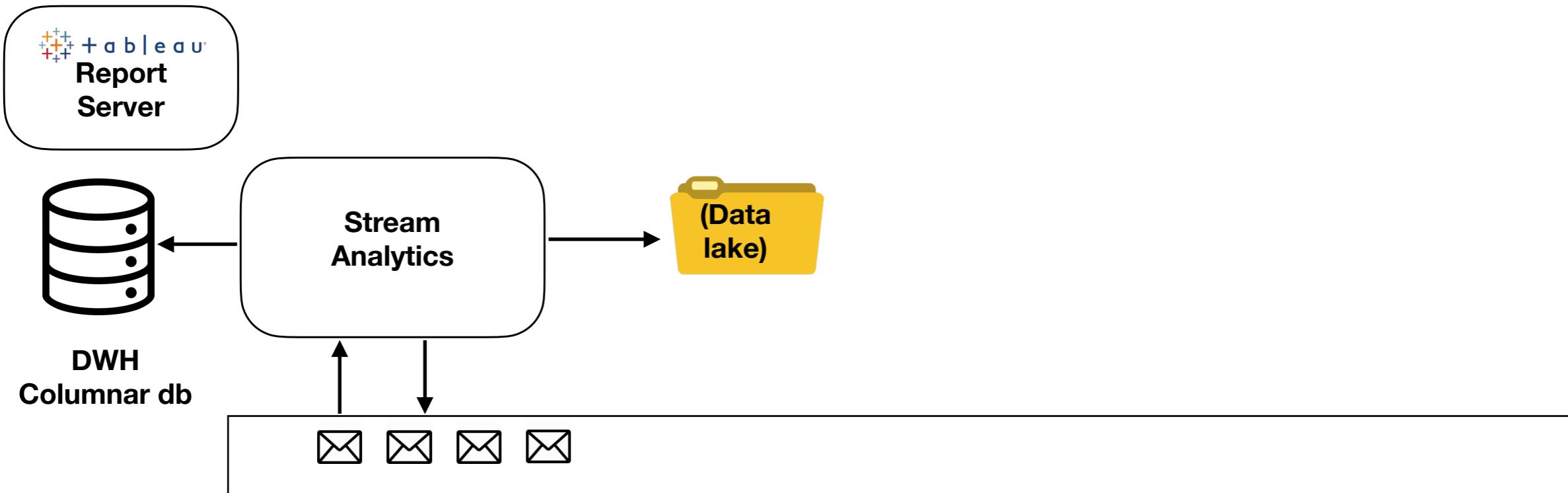


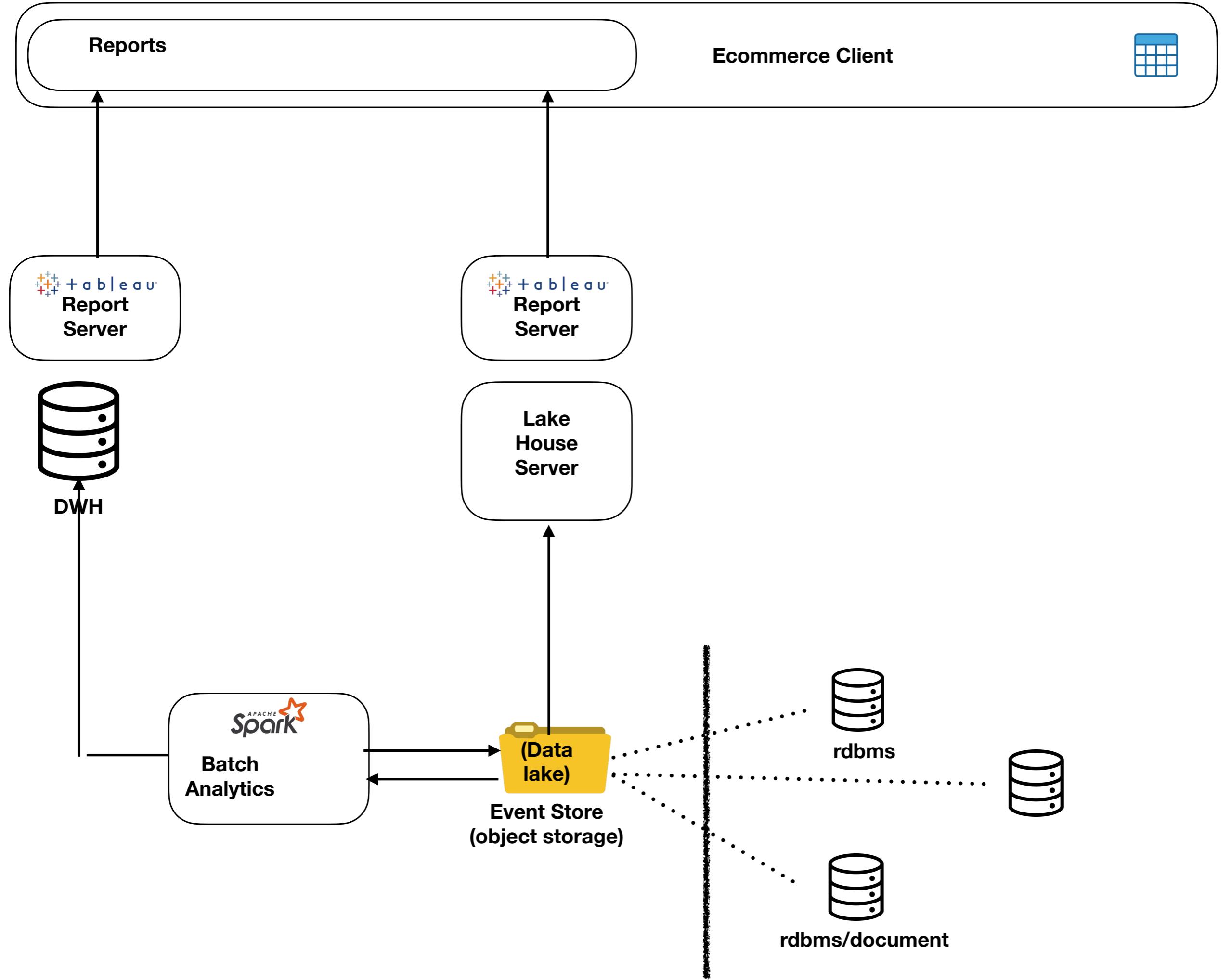




Analytics

- Lamda
- Kappa
- Lake House





TABLE_BOOK

Book ID	Genre ID	Price
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99

TABLE_GENRE

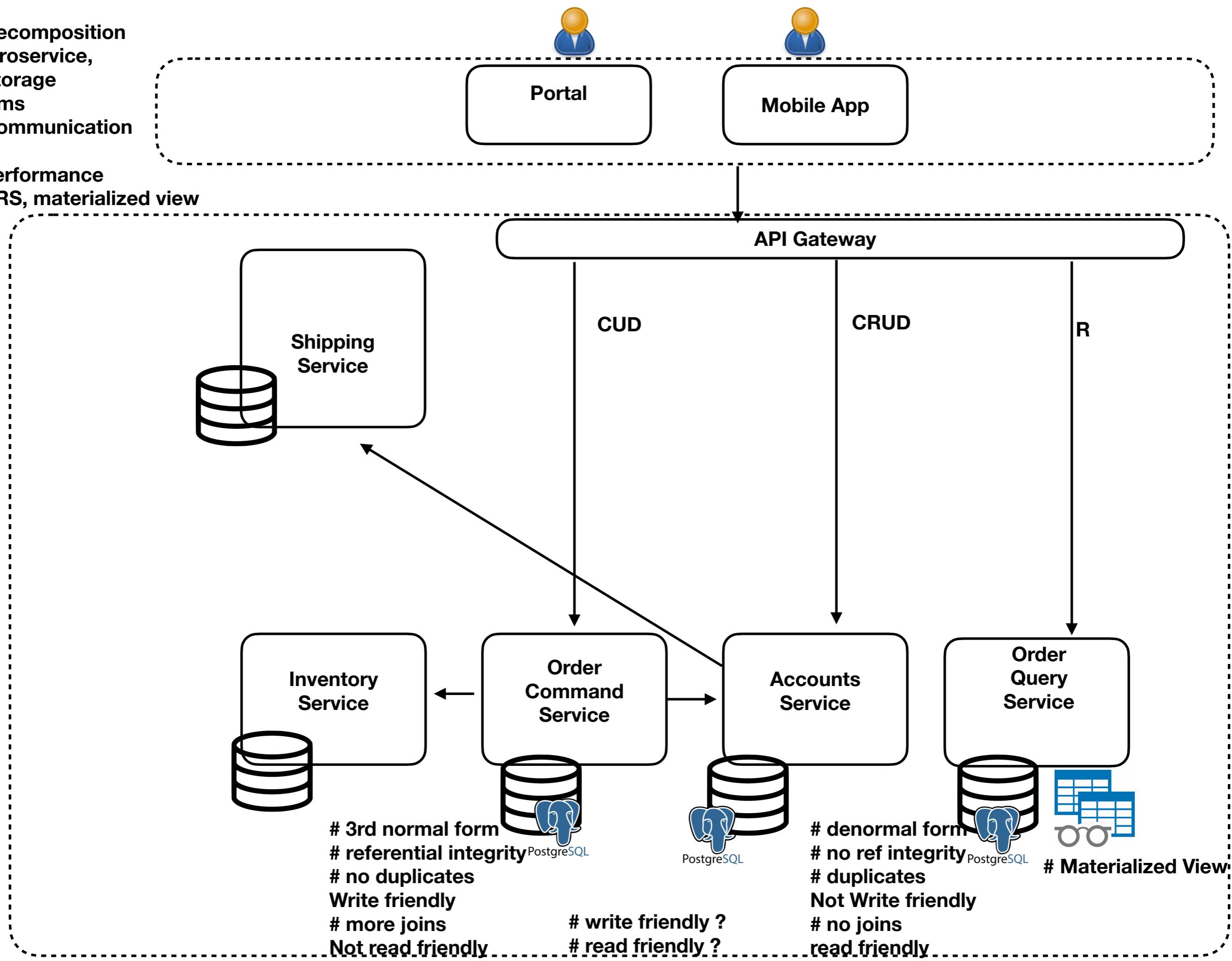
Genre ID	Genre Type
1	Gardening
2	Sports
3	Travel

3nf**# no duplicates****# joins****# not read friendly****# write friendly**

STUD_NO	STUD_NAME	STUD_STATE	STUD_COUNTRY	STUD_AGE
1	RAM	HARYANA	INDIA	20
2	RAM	PUNJAB	INDIA	19
3	SURESH	PUNJAB	INDIA	21

Table 4**dnf****# duplicates****# no joins****# read friendly****# not write friendly**

Decomposition
microservice,
Storage
rdbms
Communication
?
performance
CQRS, materialized view



Decomposition

microservice,

Storage

rdbms

Communication

Topic /hub

performance

CQRS, materialized view

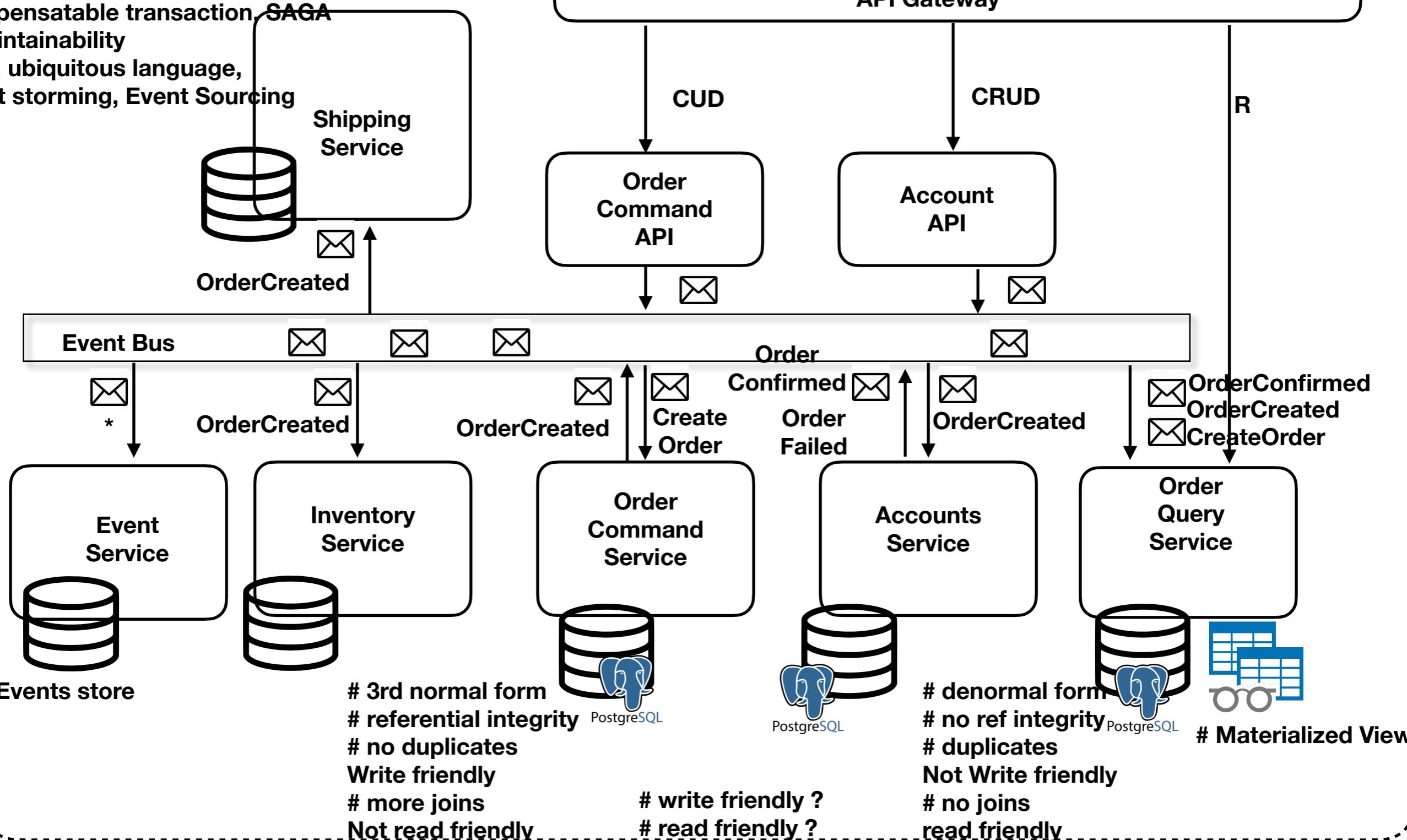
reliability

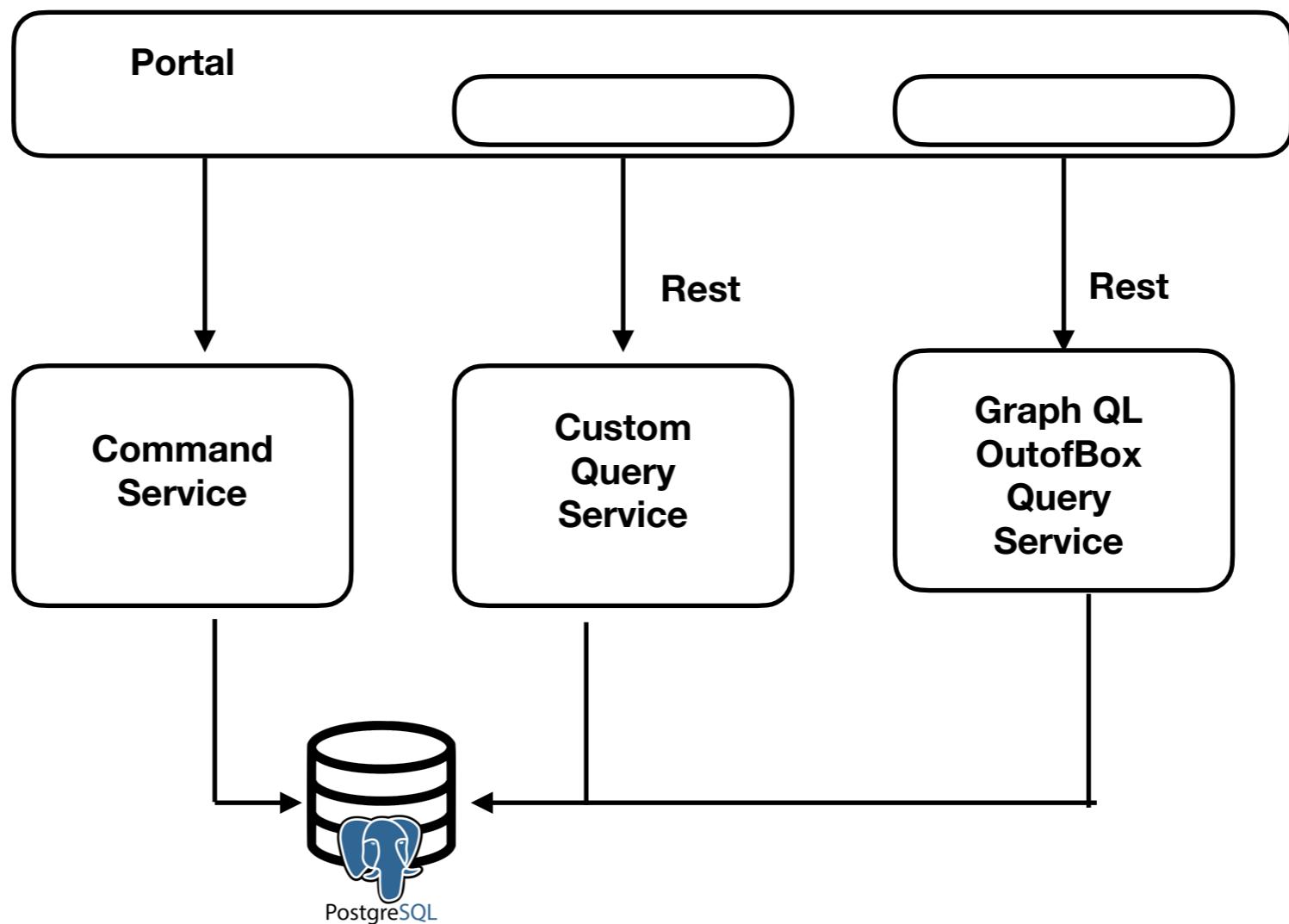
Compensatable transaction, SAGA

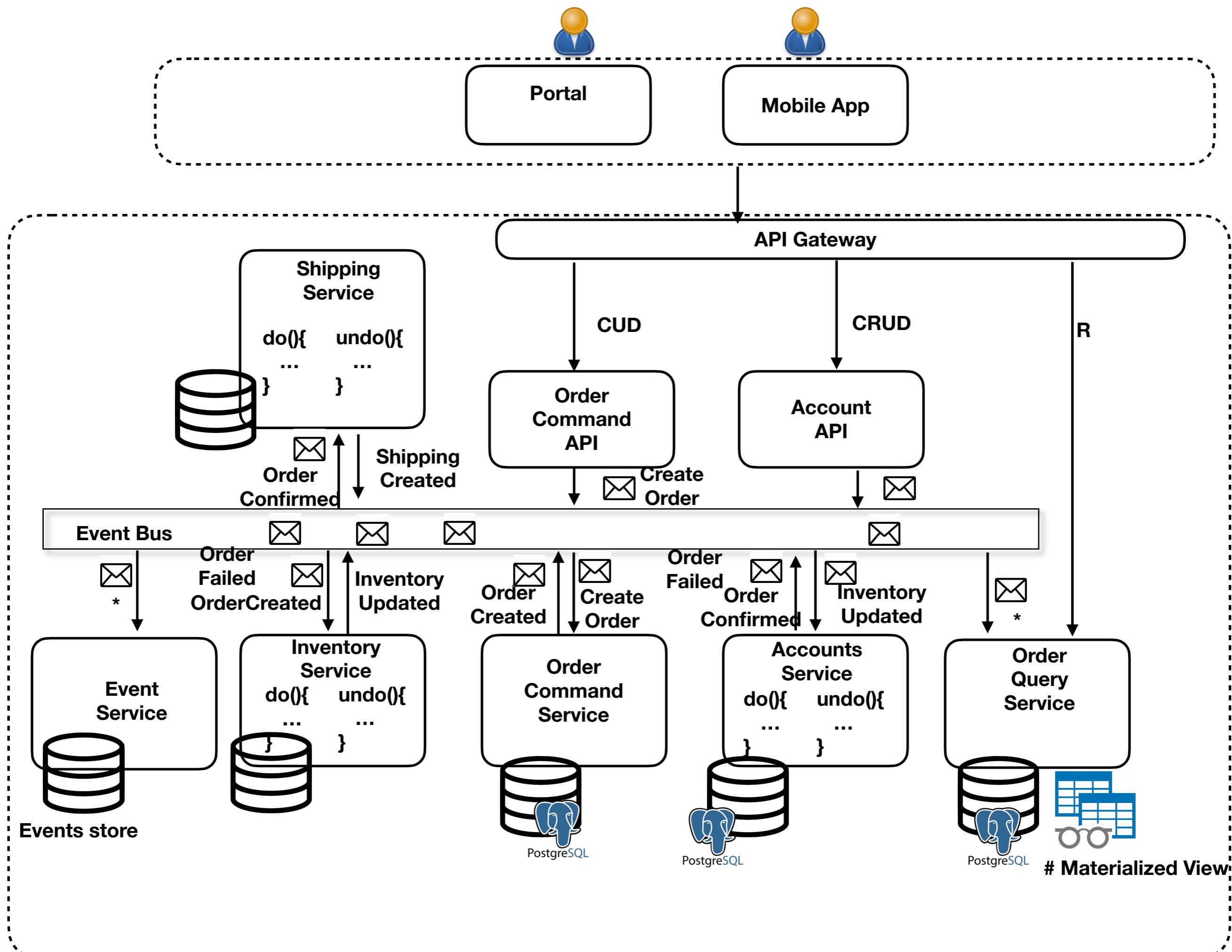
Maintainability

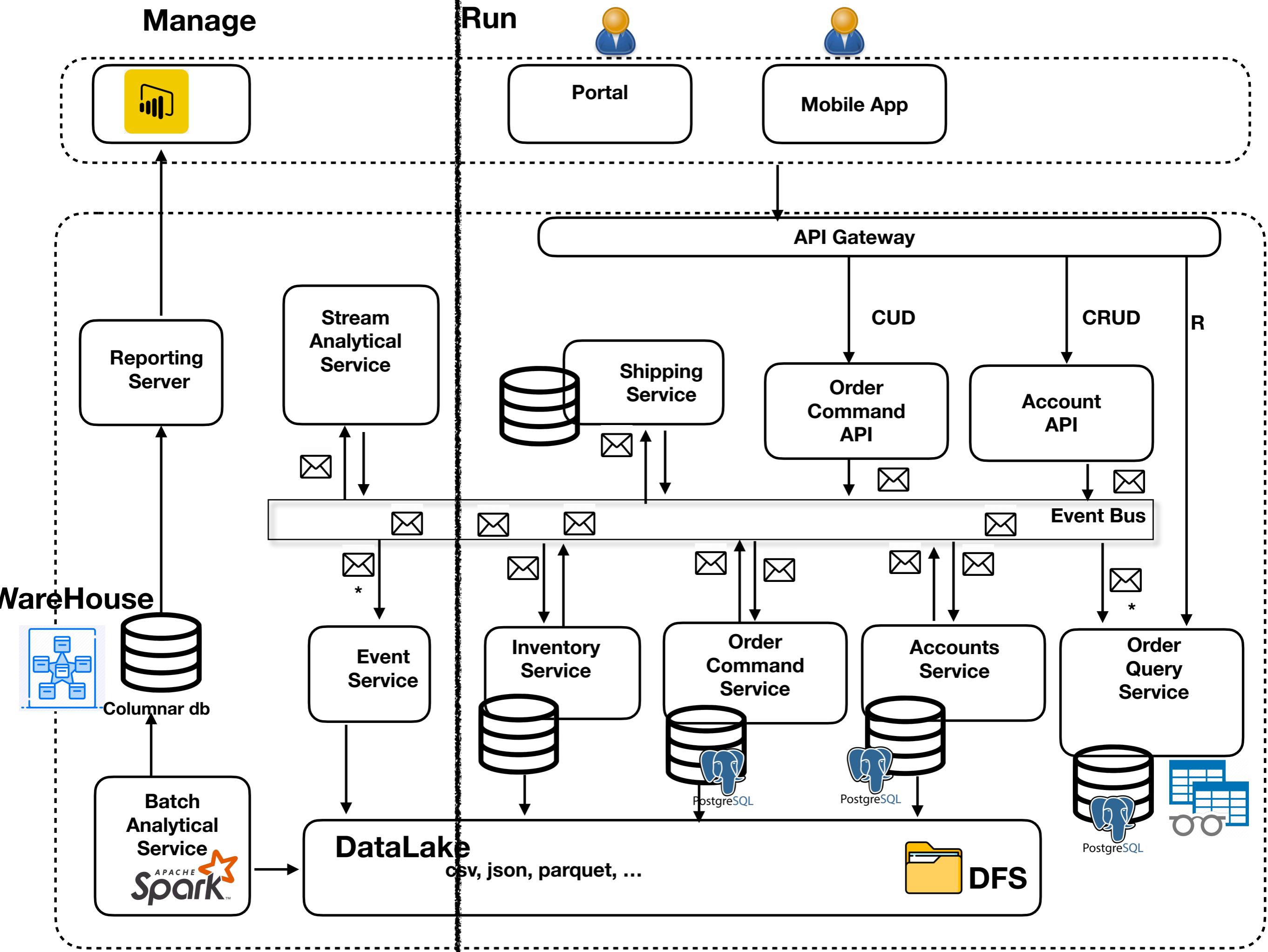
DDP, ubiquitous language,

Event storming, Event Sourcing



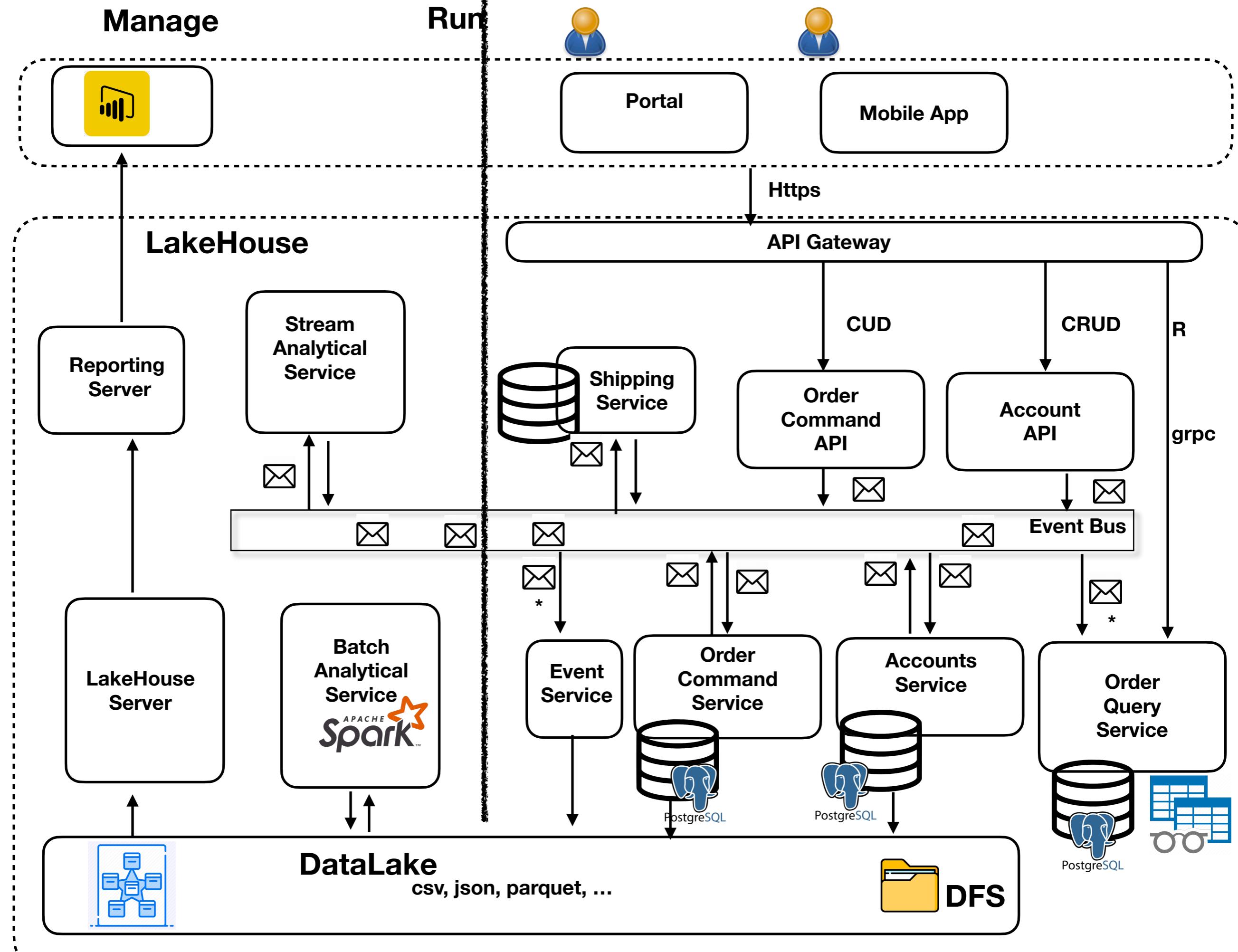






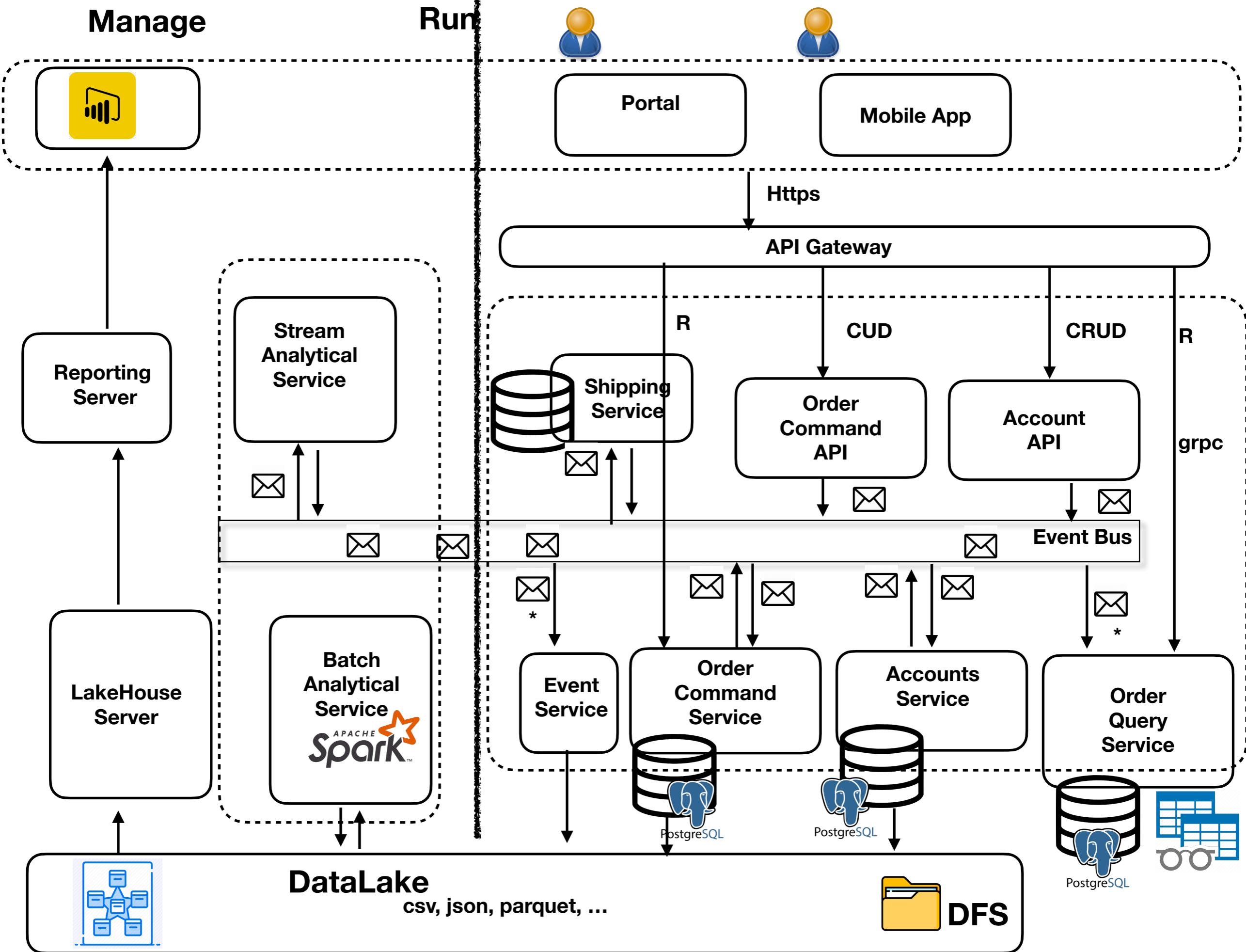
Manage

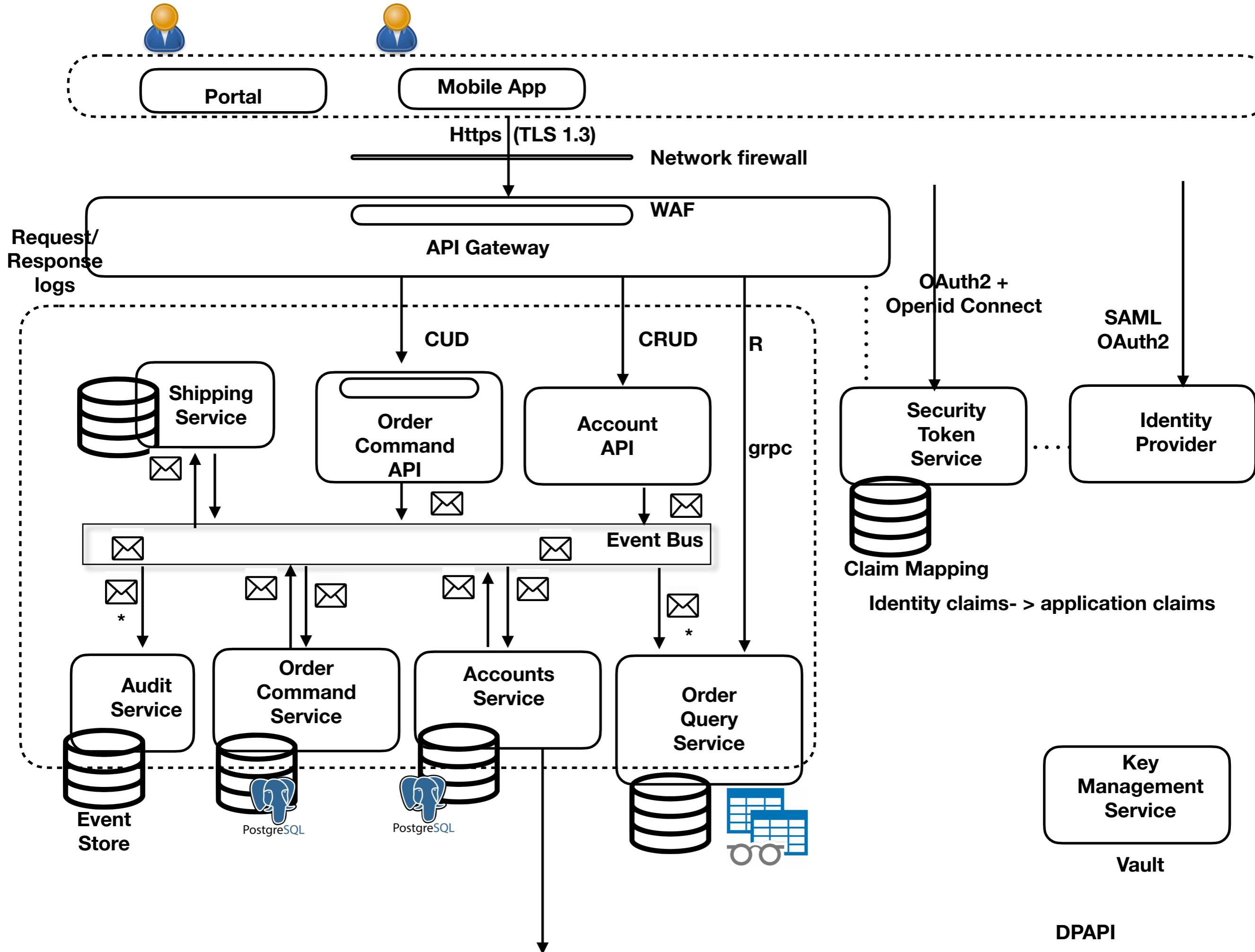
Run



Manage

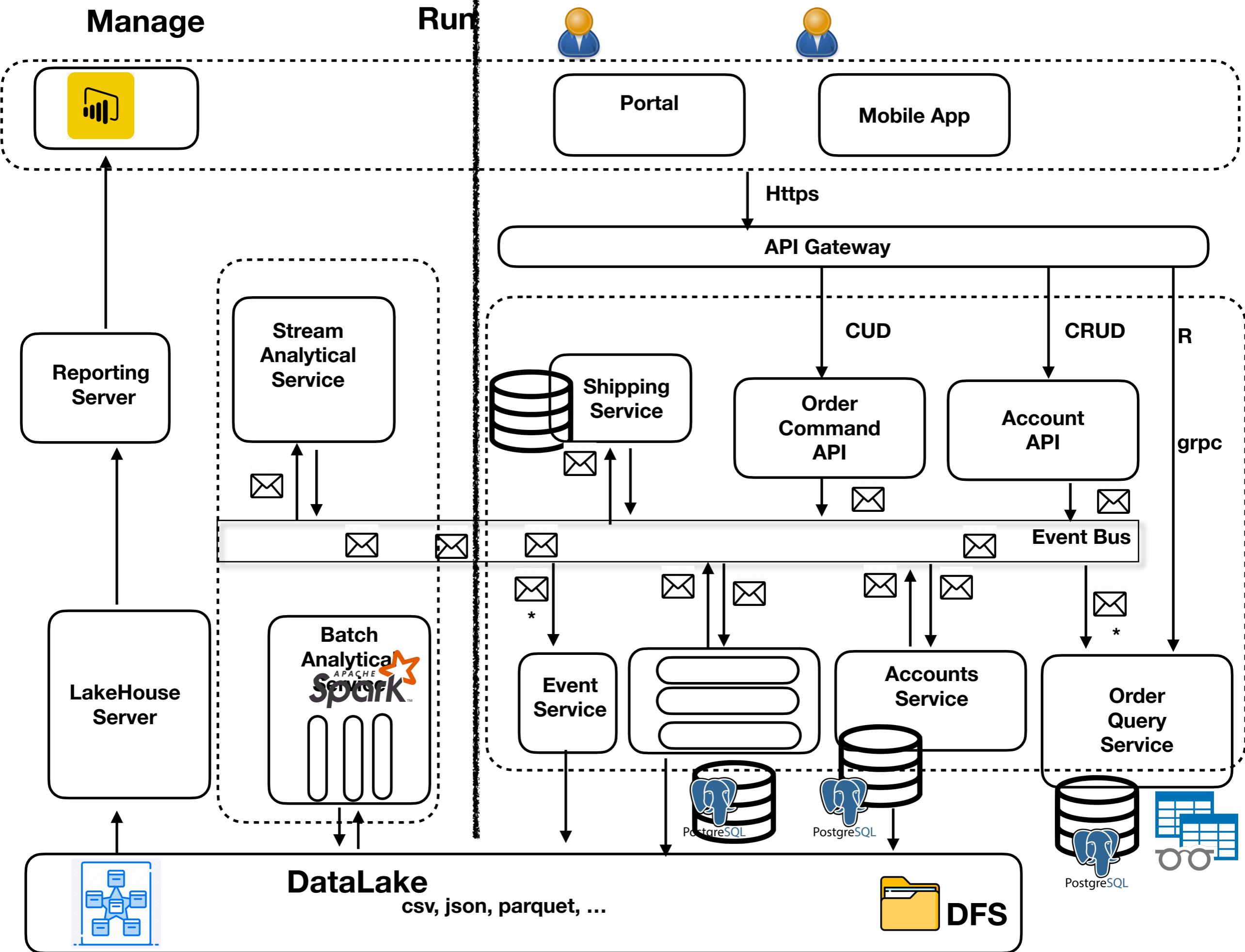
Run

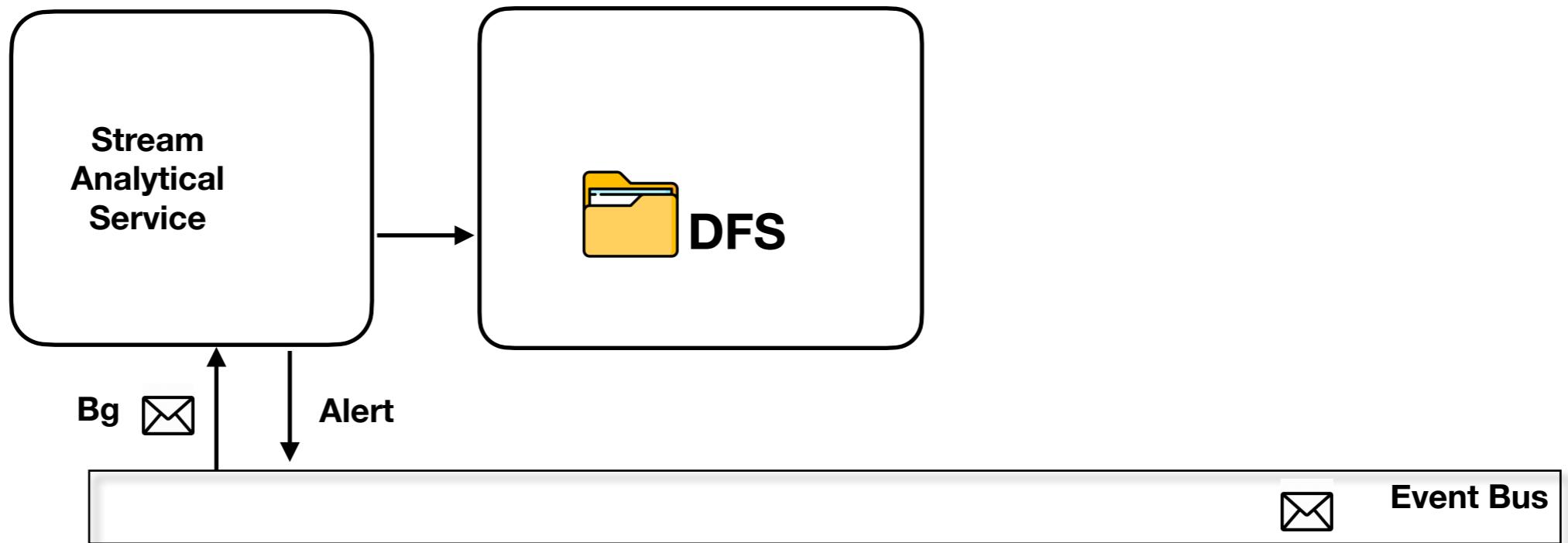




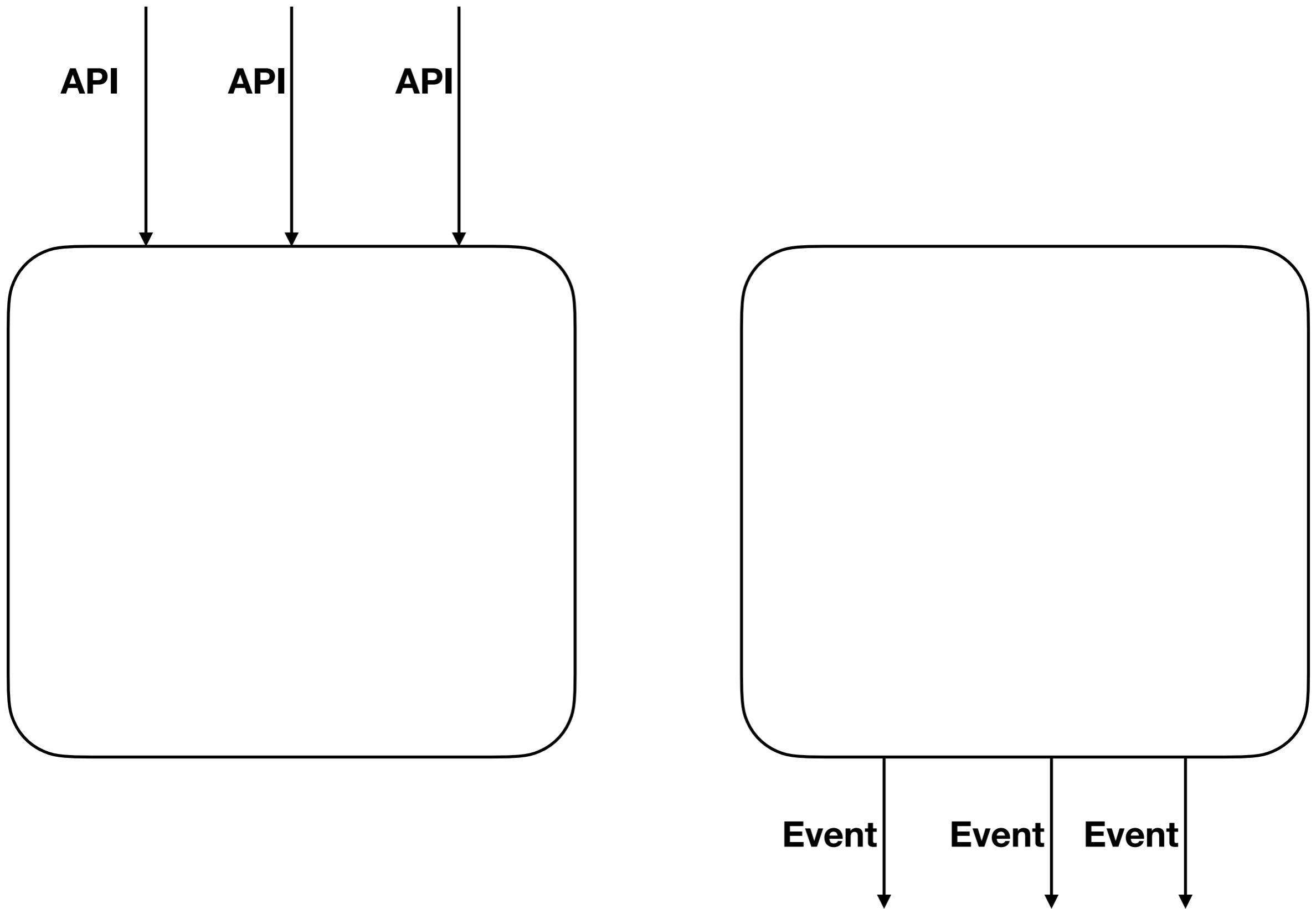
Manage

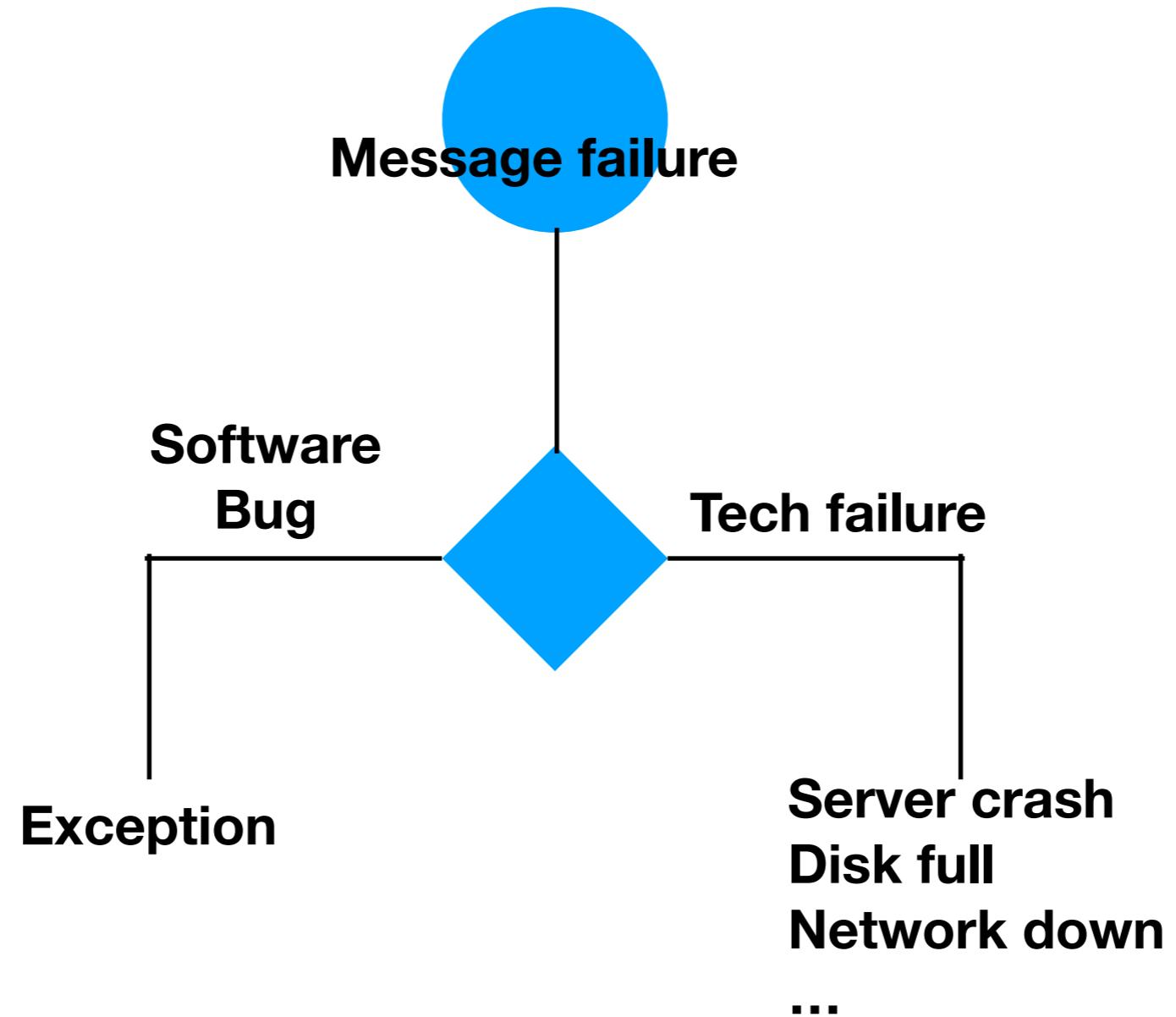
Run





- select sum(salary) from emp
- emp_collection.sum()
-







Bidding Application

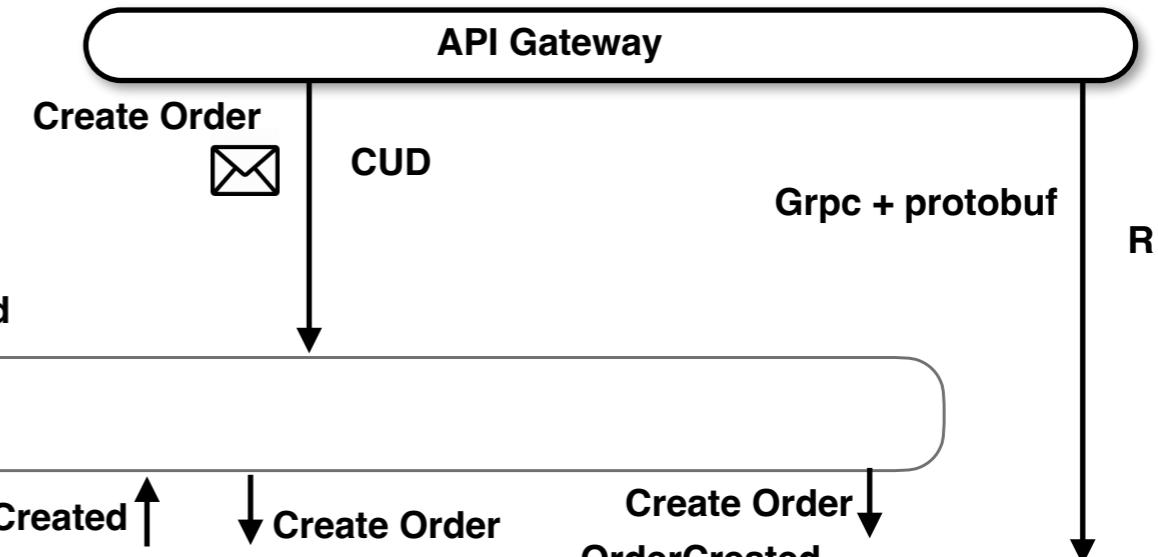
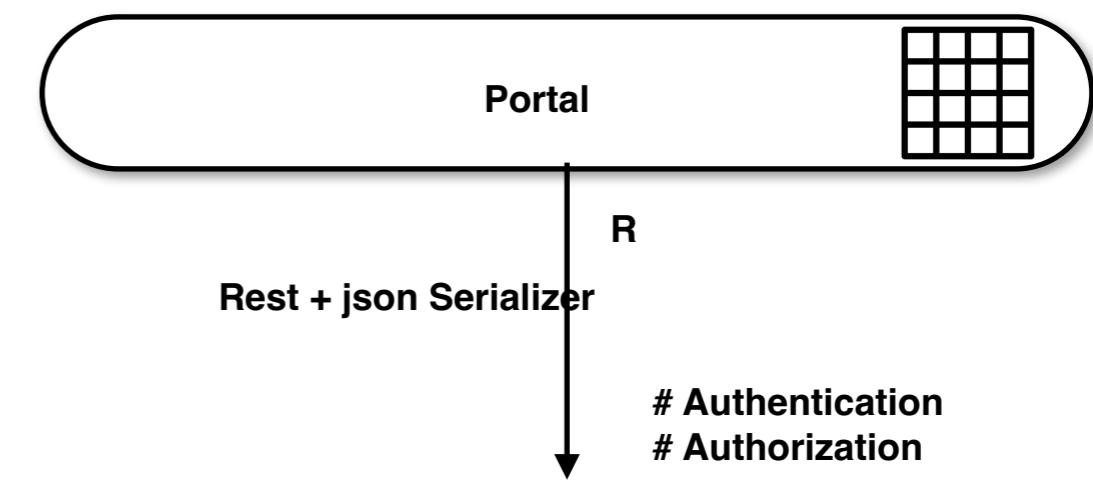
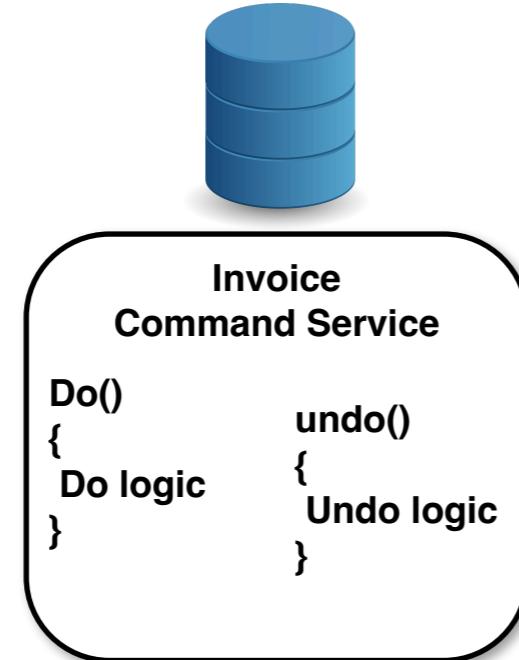


```

# CQRS
# Materialized View
# SAGA (compensatable transaction)
# Eventual Consistency
# CAP
# Event sourcing (DDD)
# ubiquitous language
# EDA
# Choreography
# API Gateway
# Event & Command

```

Stream Analytics
(Look for patterns in < 7 days)

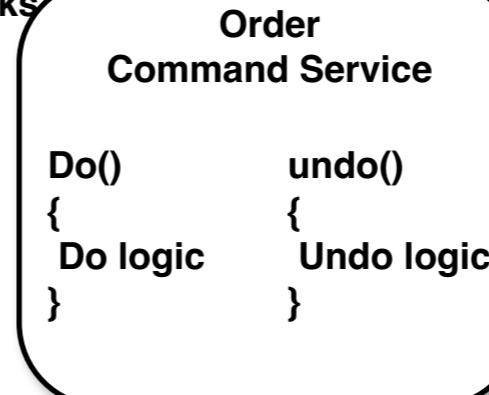


Domain Events & Domain Commands



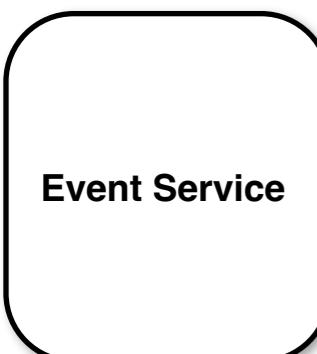
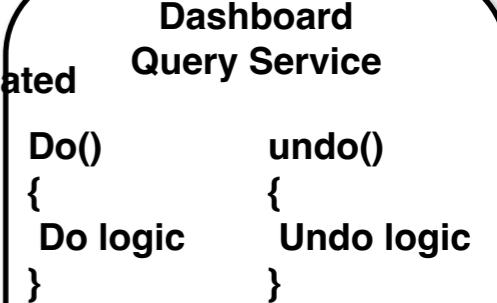
OrderCreated ↑ ↓ Create Order

No Stocks



Create Order
↓
OrderCreated

No Stocks
InvoiceCreated



(Look for patterns for years)

DWH Analytics



Big Data Analytics

(Look for patterns for years)



Snapshot

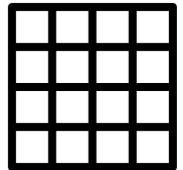
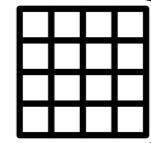
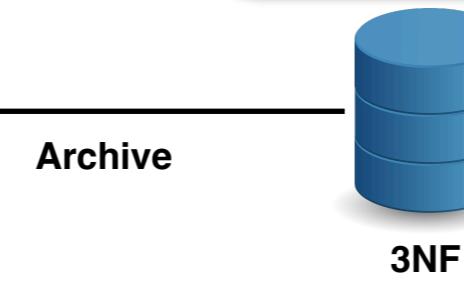


Invoice
Command Service

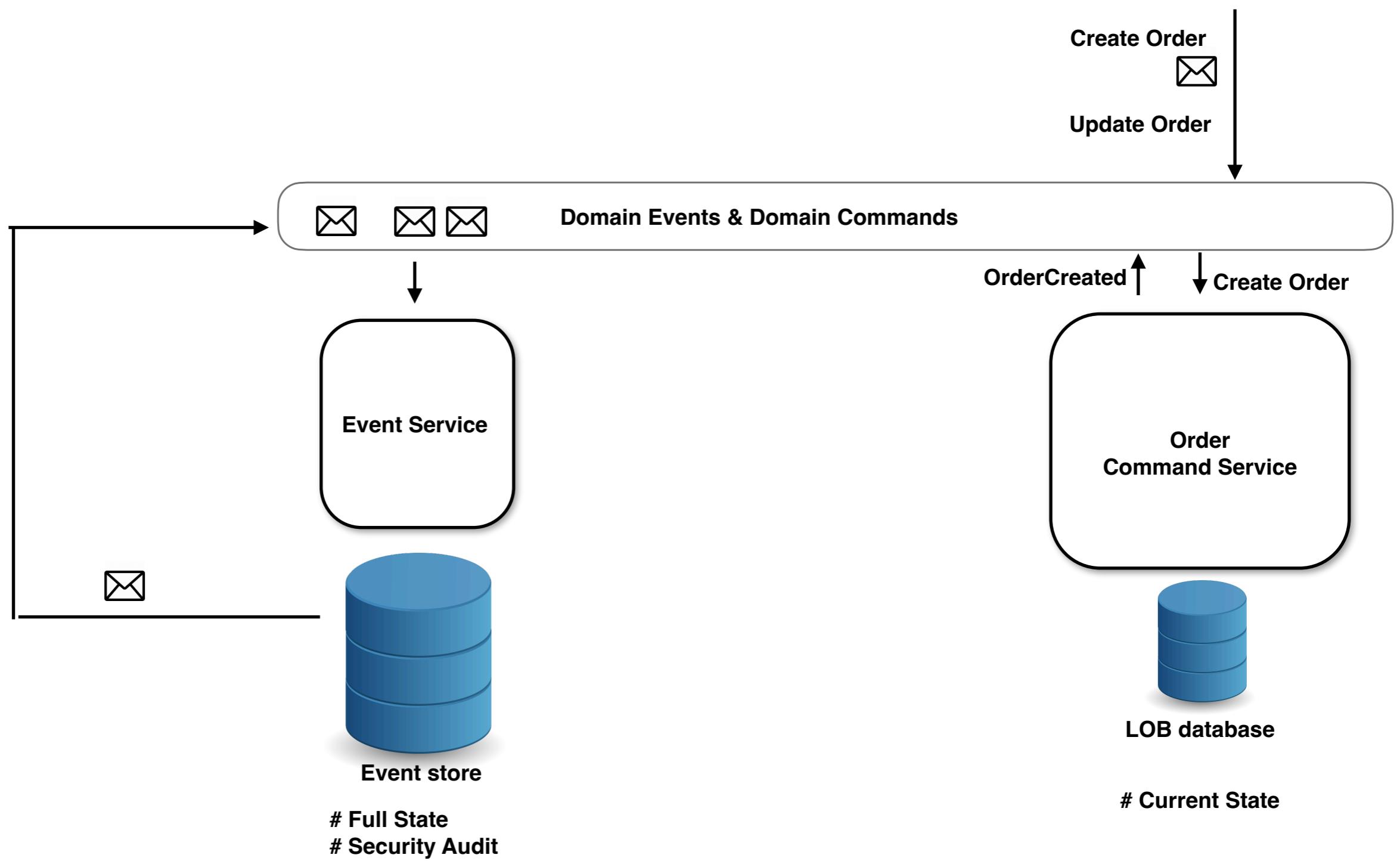
```

Do()
{
    Do logic
}
undo()
{
    Undo logic
}

```



Full State
Security Audit





Buyer

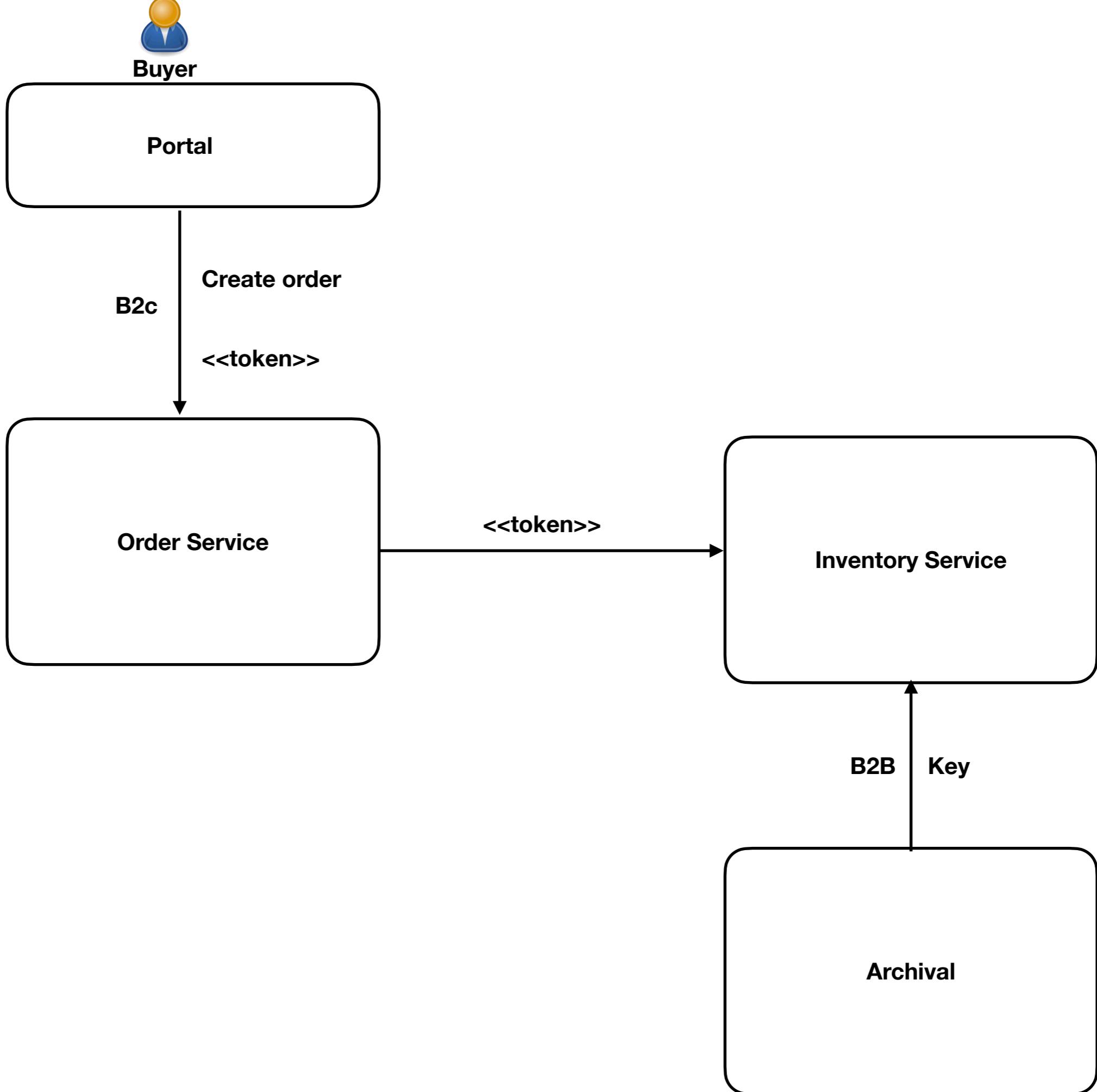
Portal

Create order

Order Service

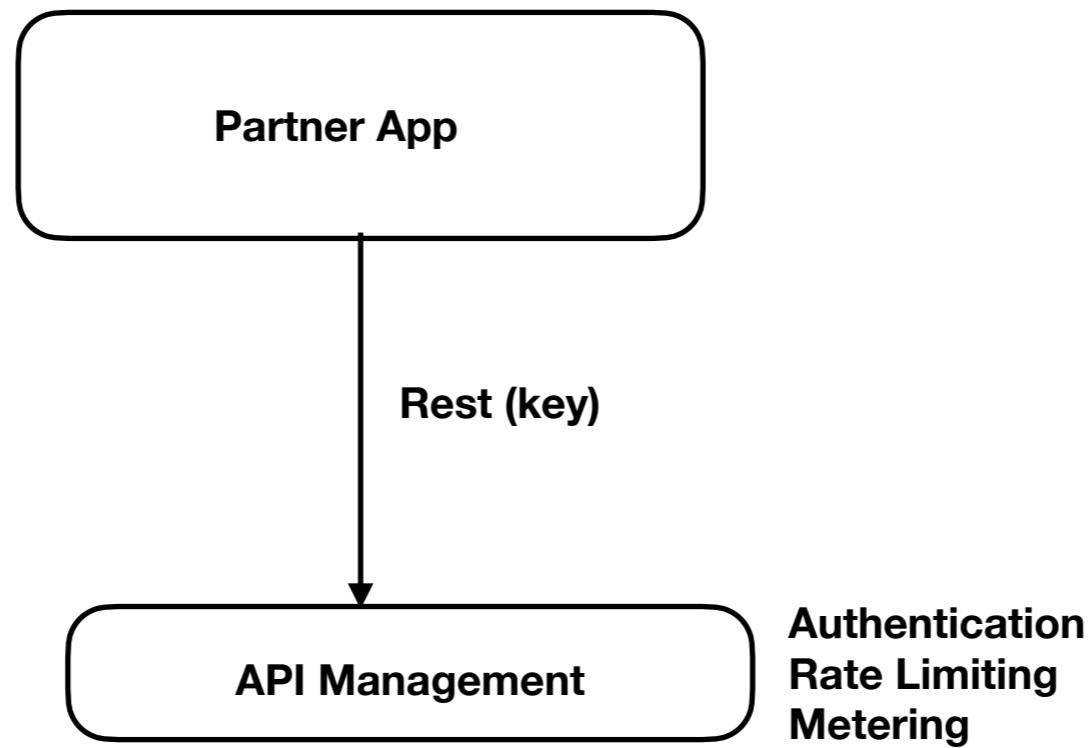
Catalogue Service



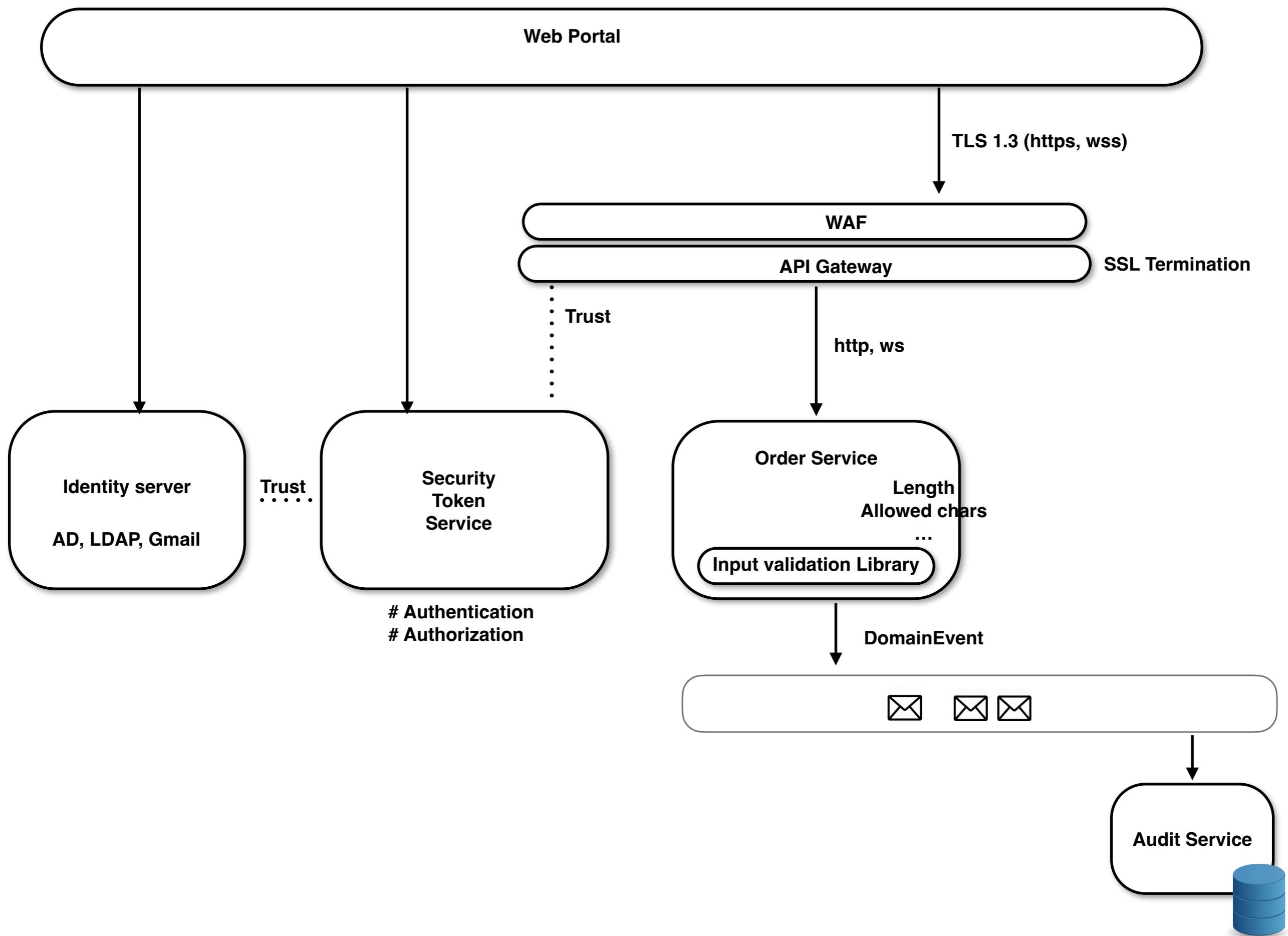




Buyer

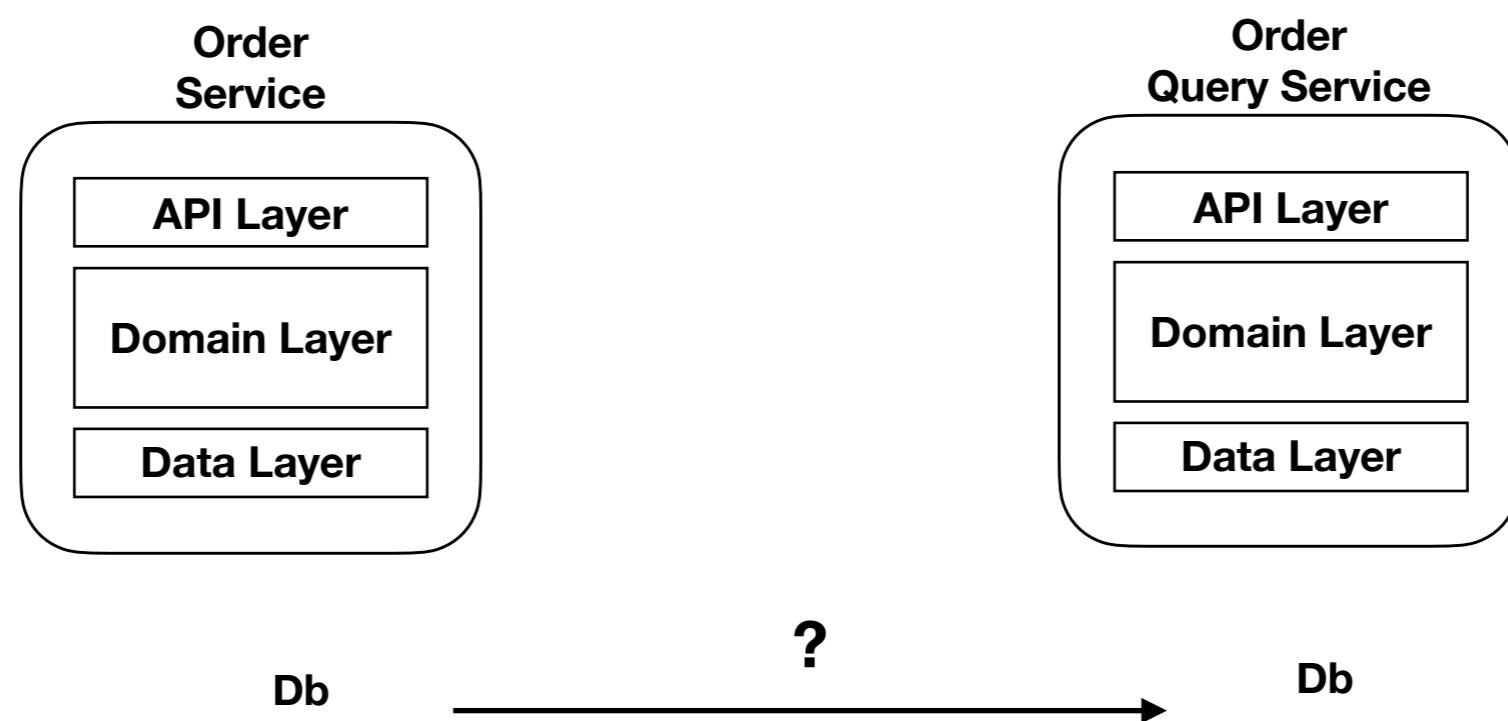
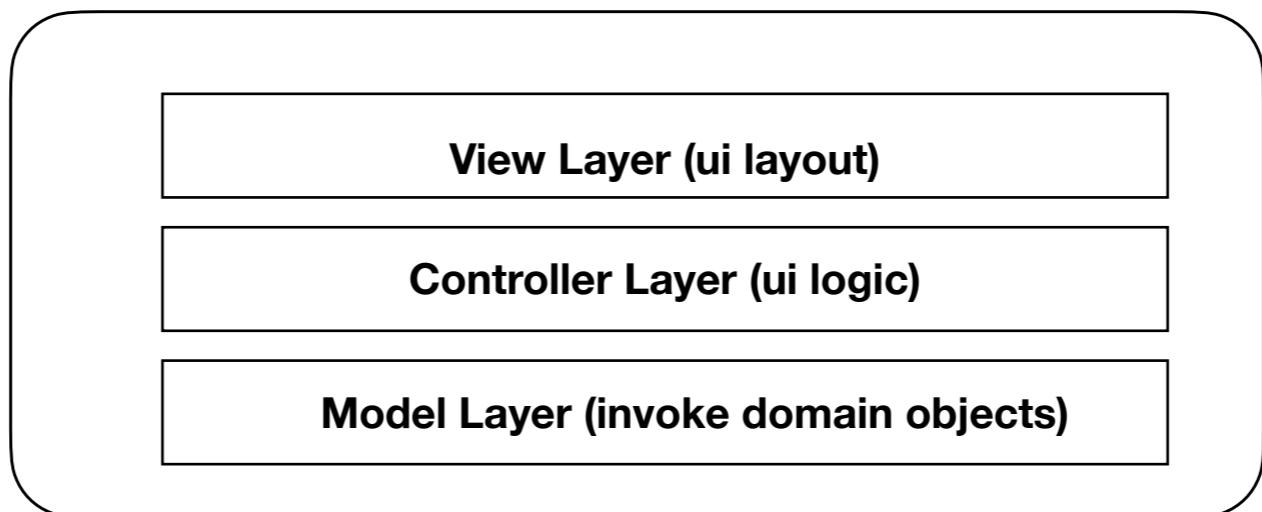


Enterprise Data

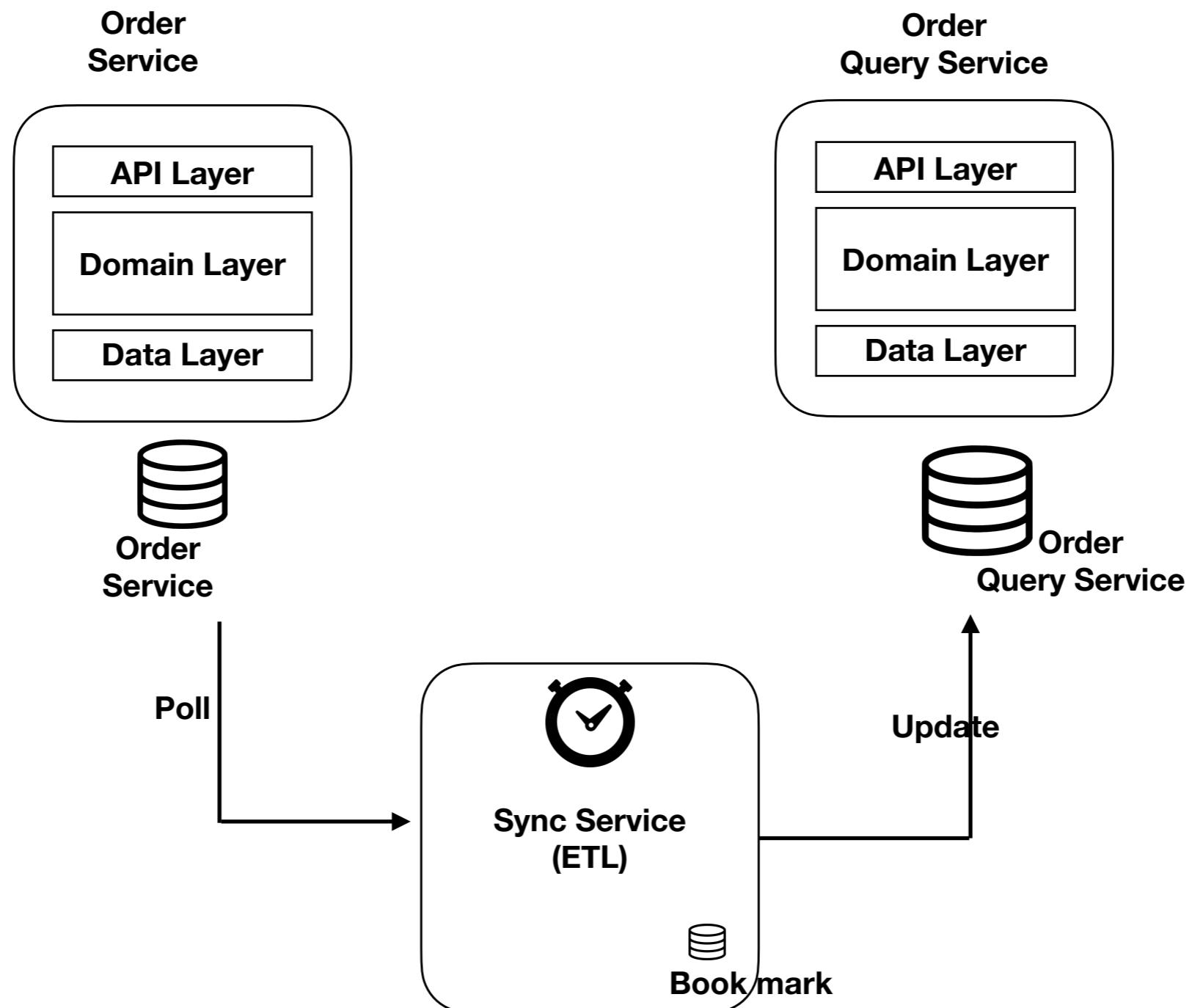


View - CDC

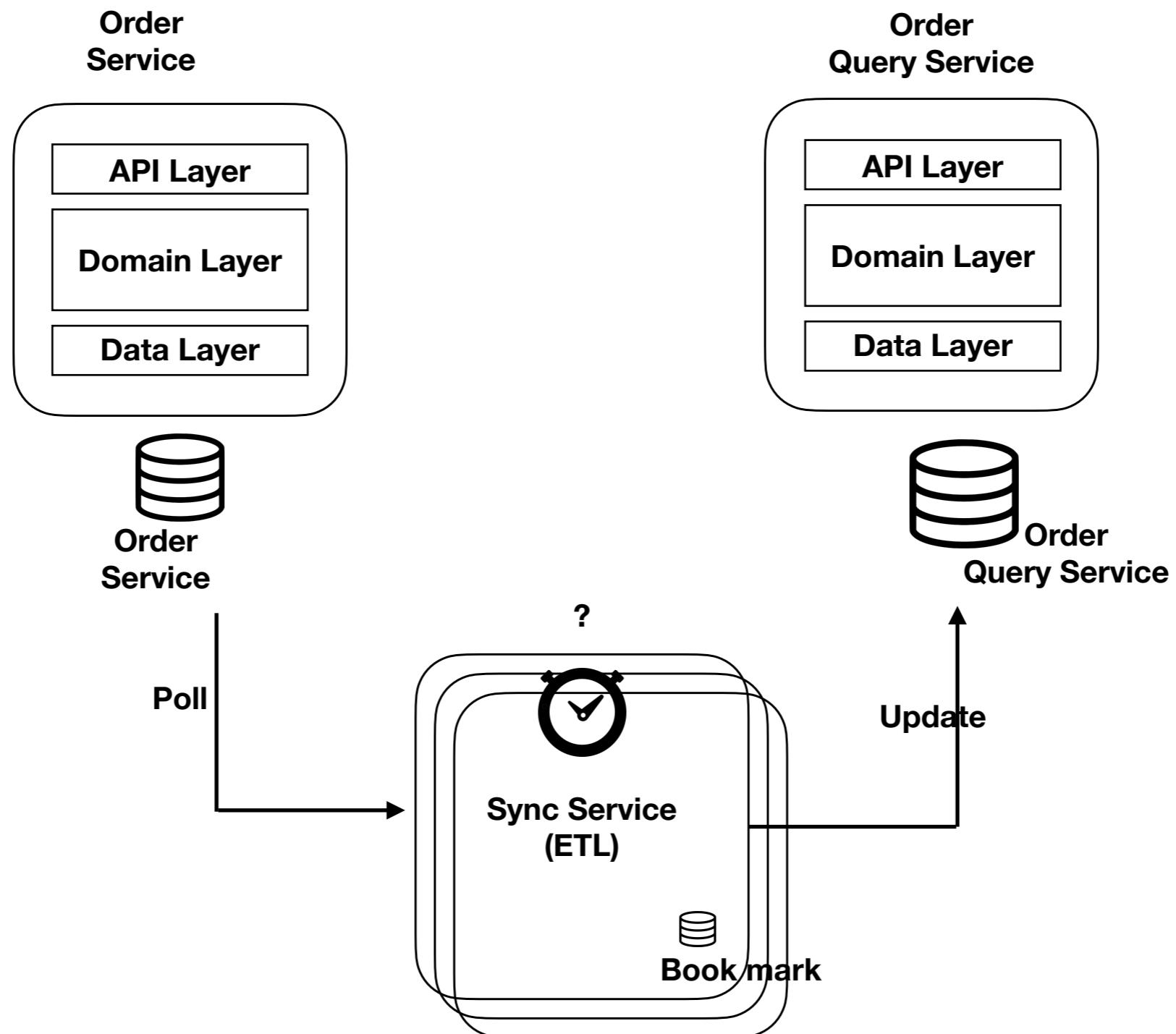
Web Portal



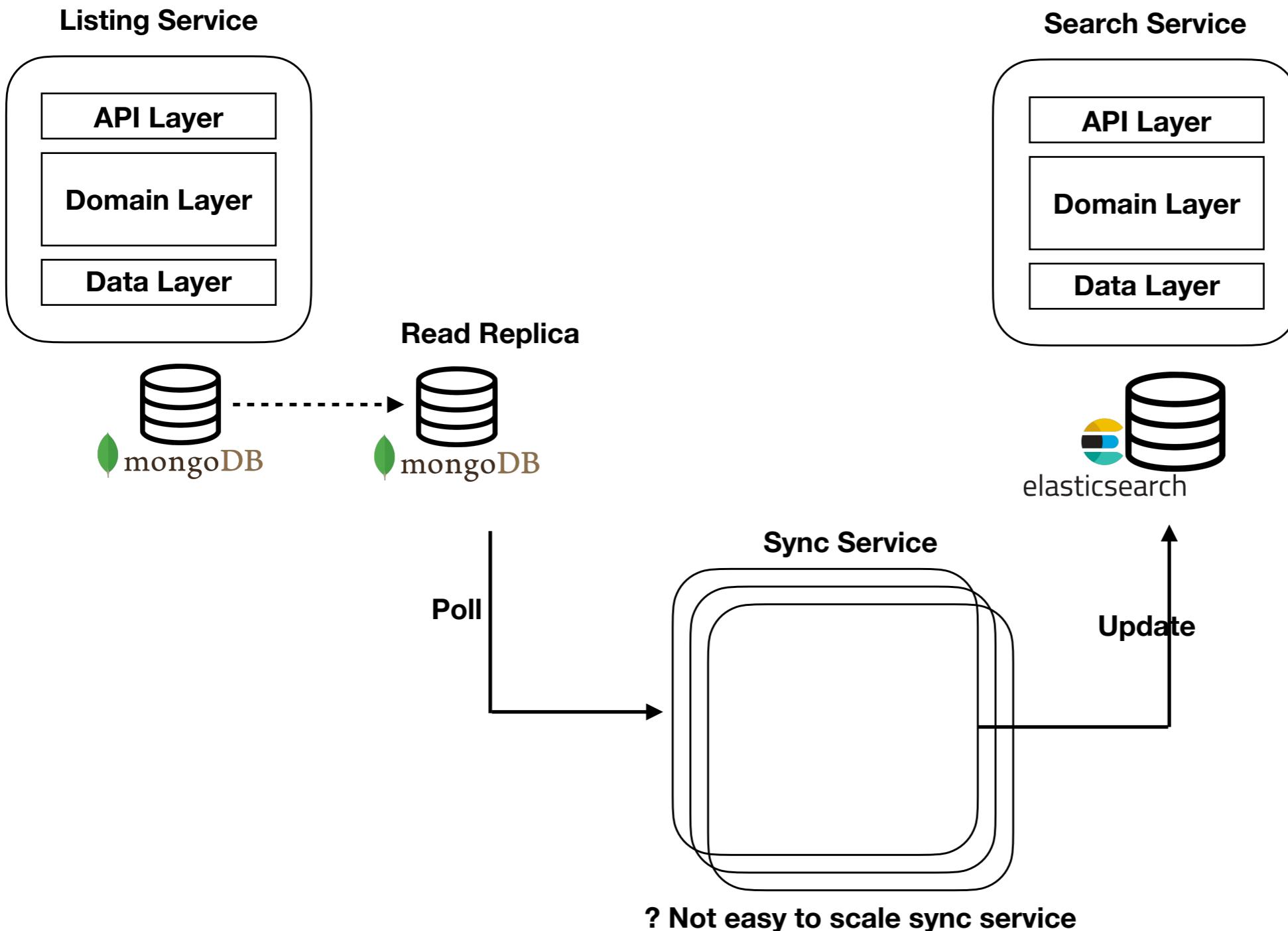
Design 1 - poll



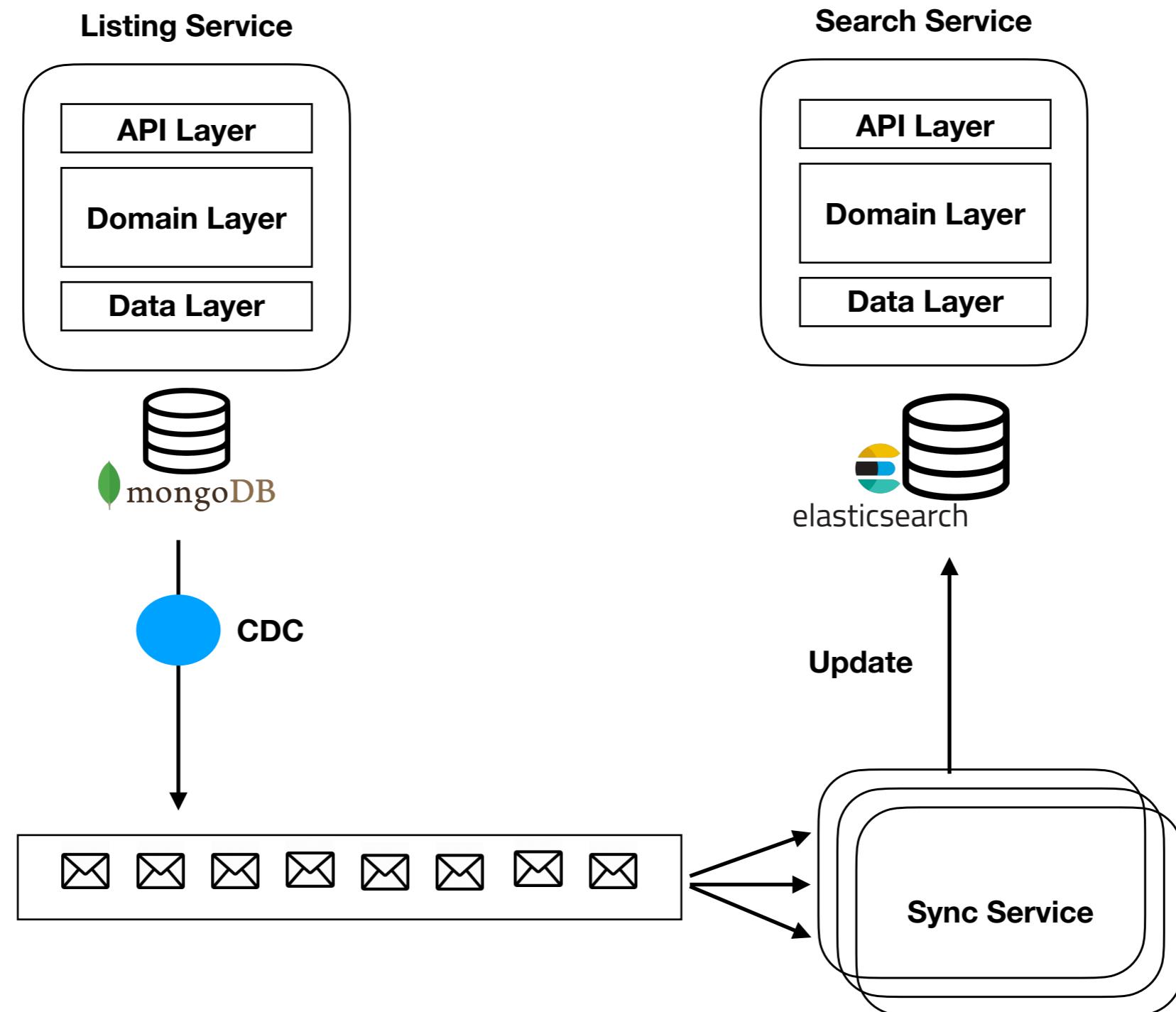
Design 1 - poll

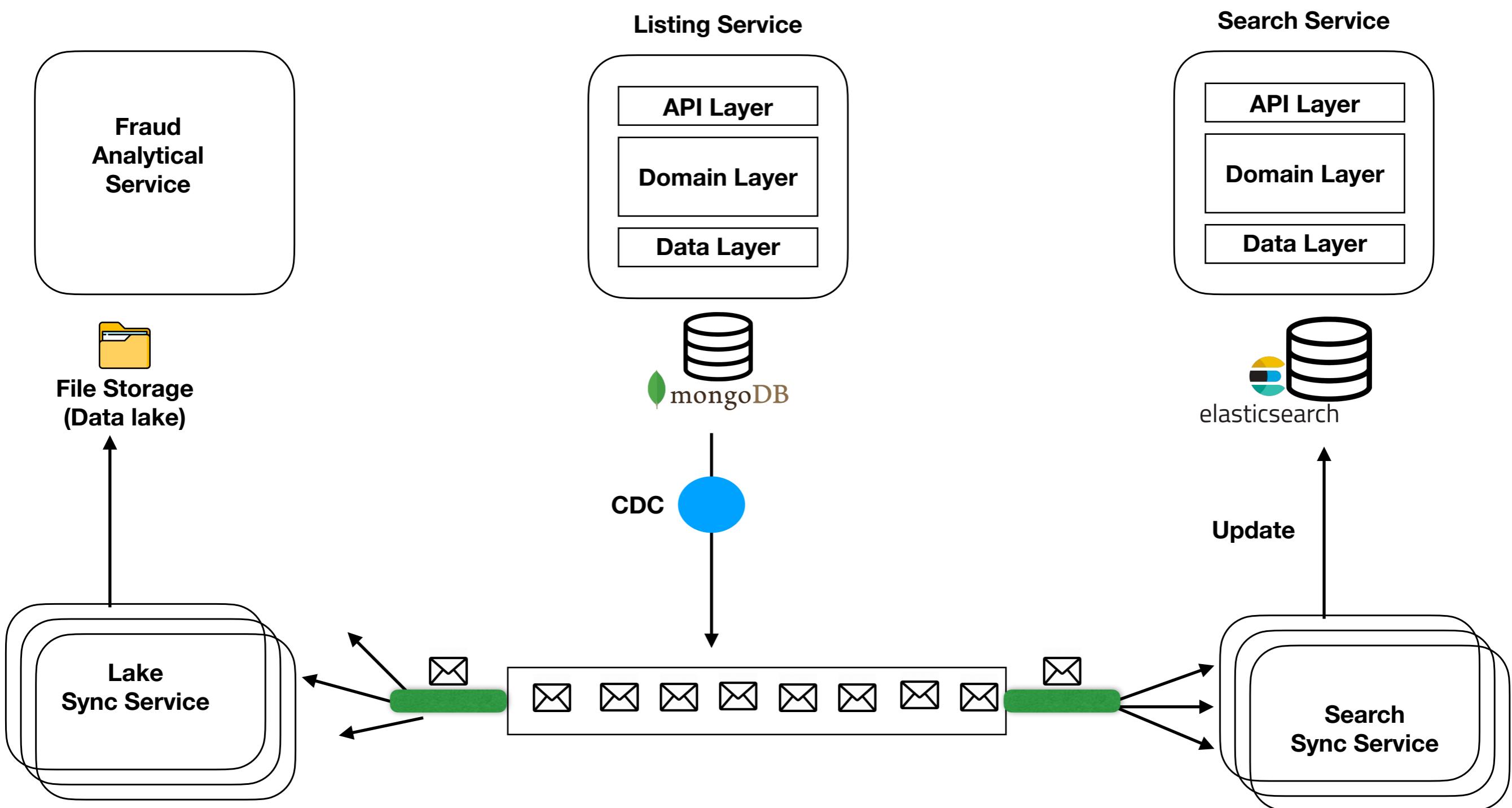


Design 2 - Read Replica poll



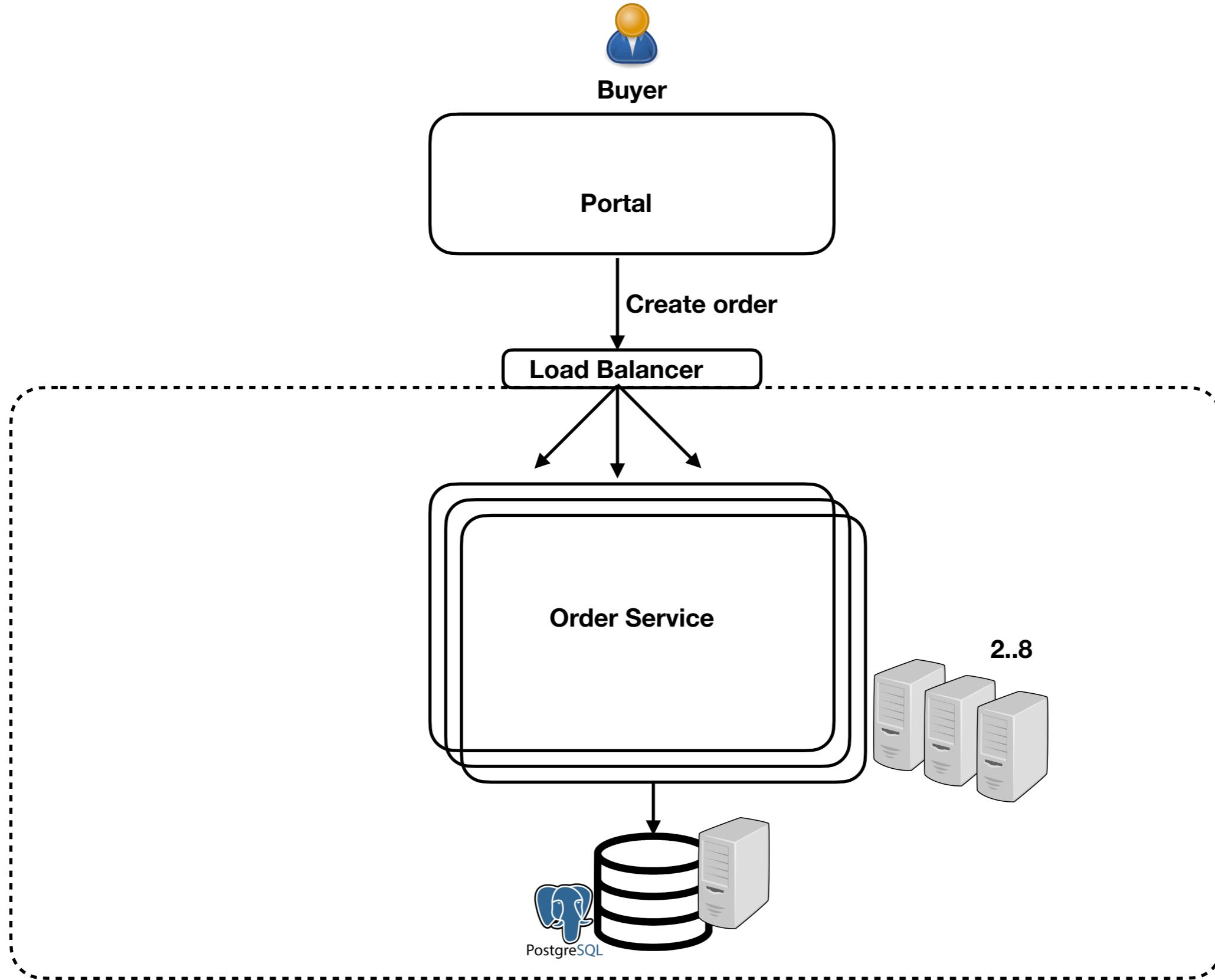
Design 3 - CDC





Deployment View

#

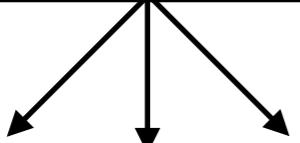


Portal

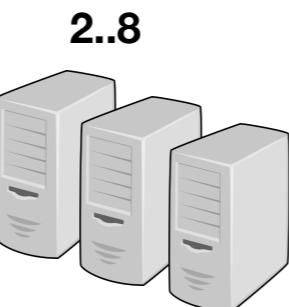
↓

Create order

Load Balancer



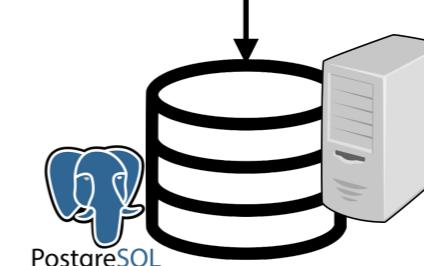
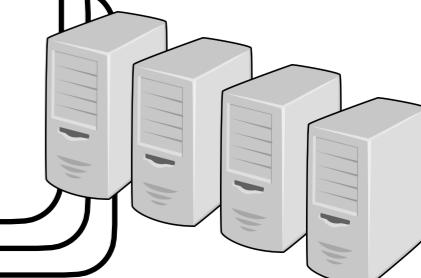
Order Command Service



Create order

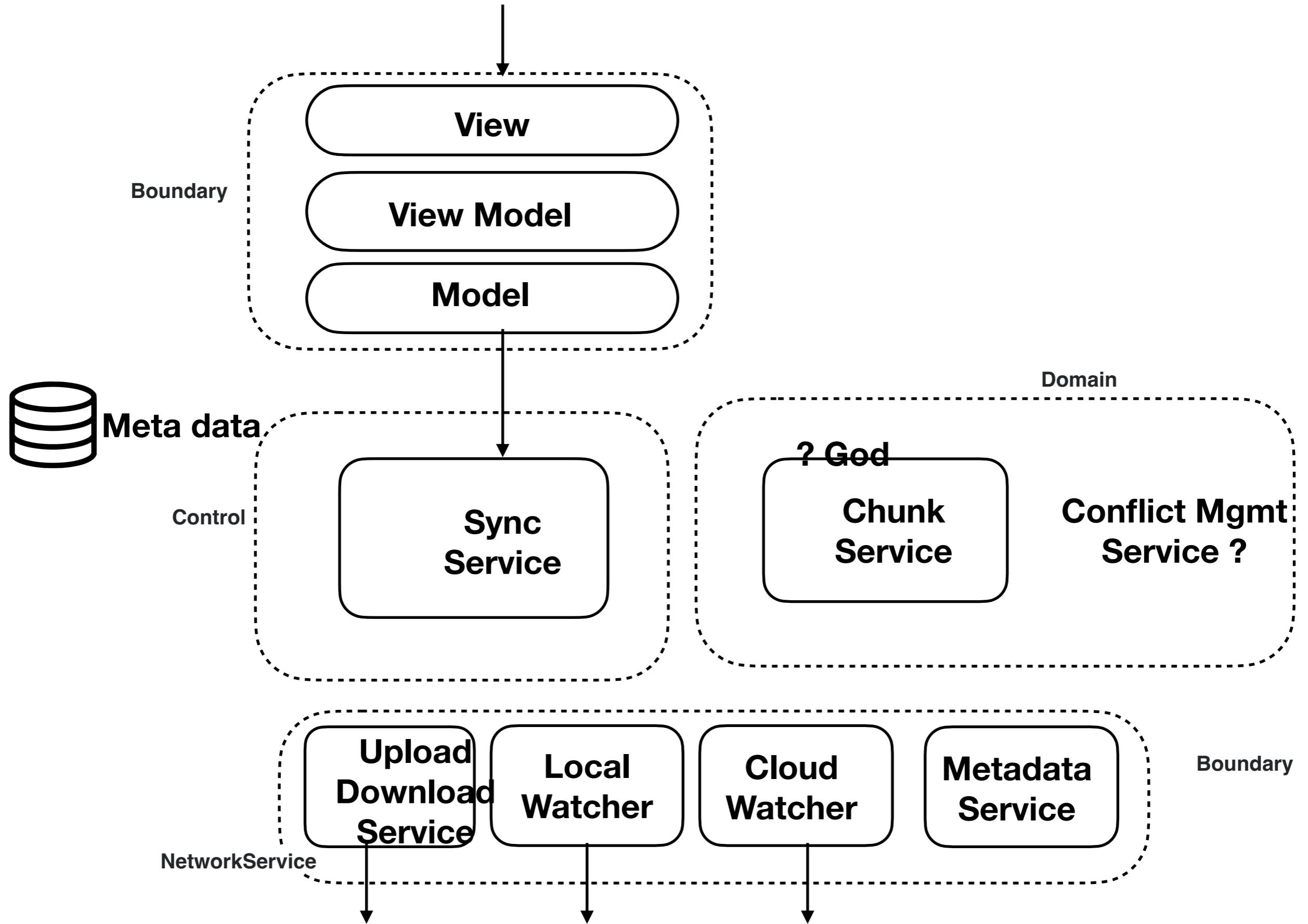


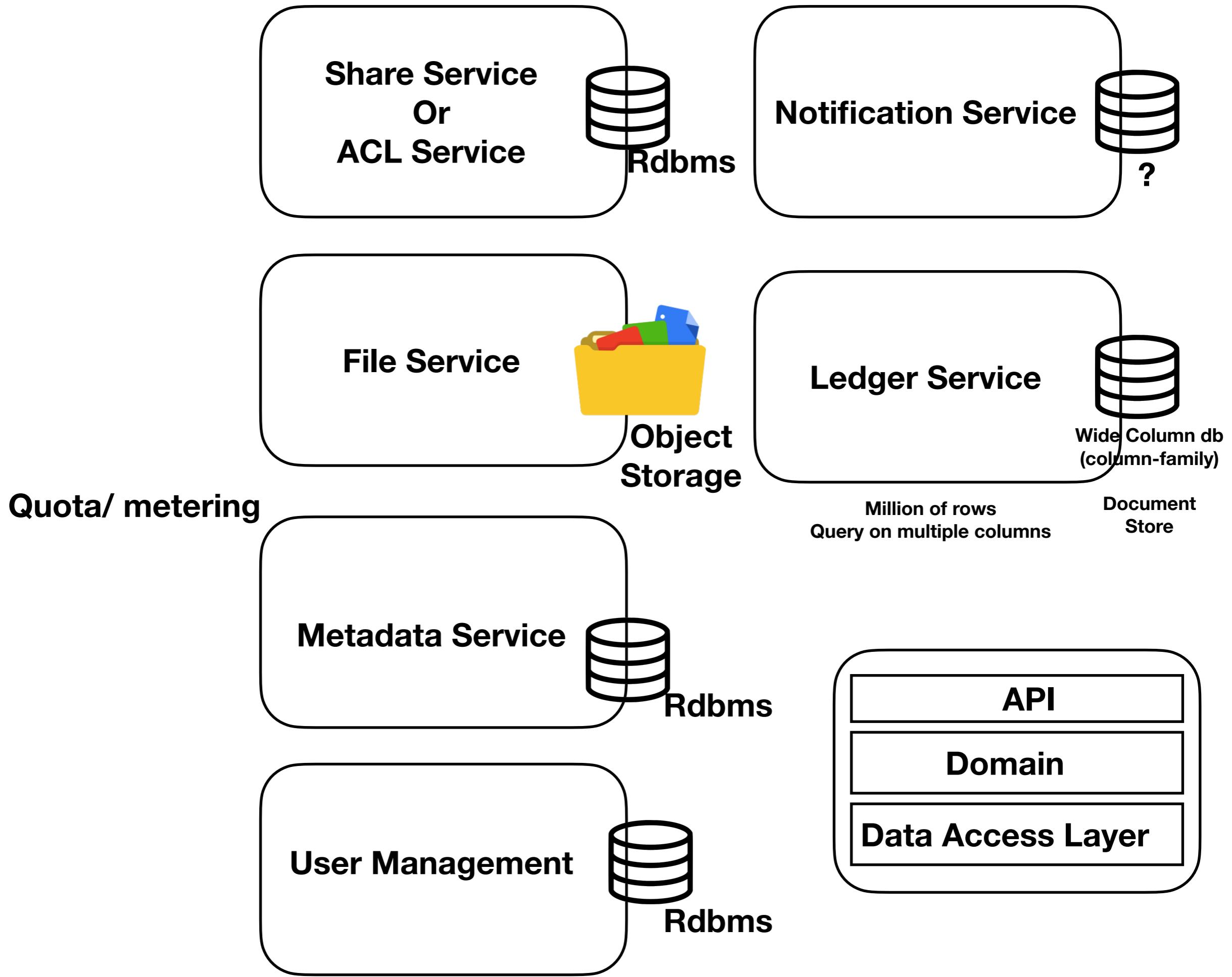
Order Service

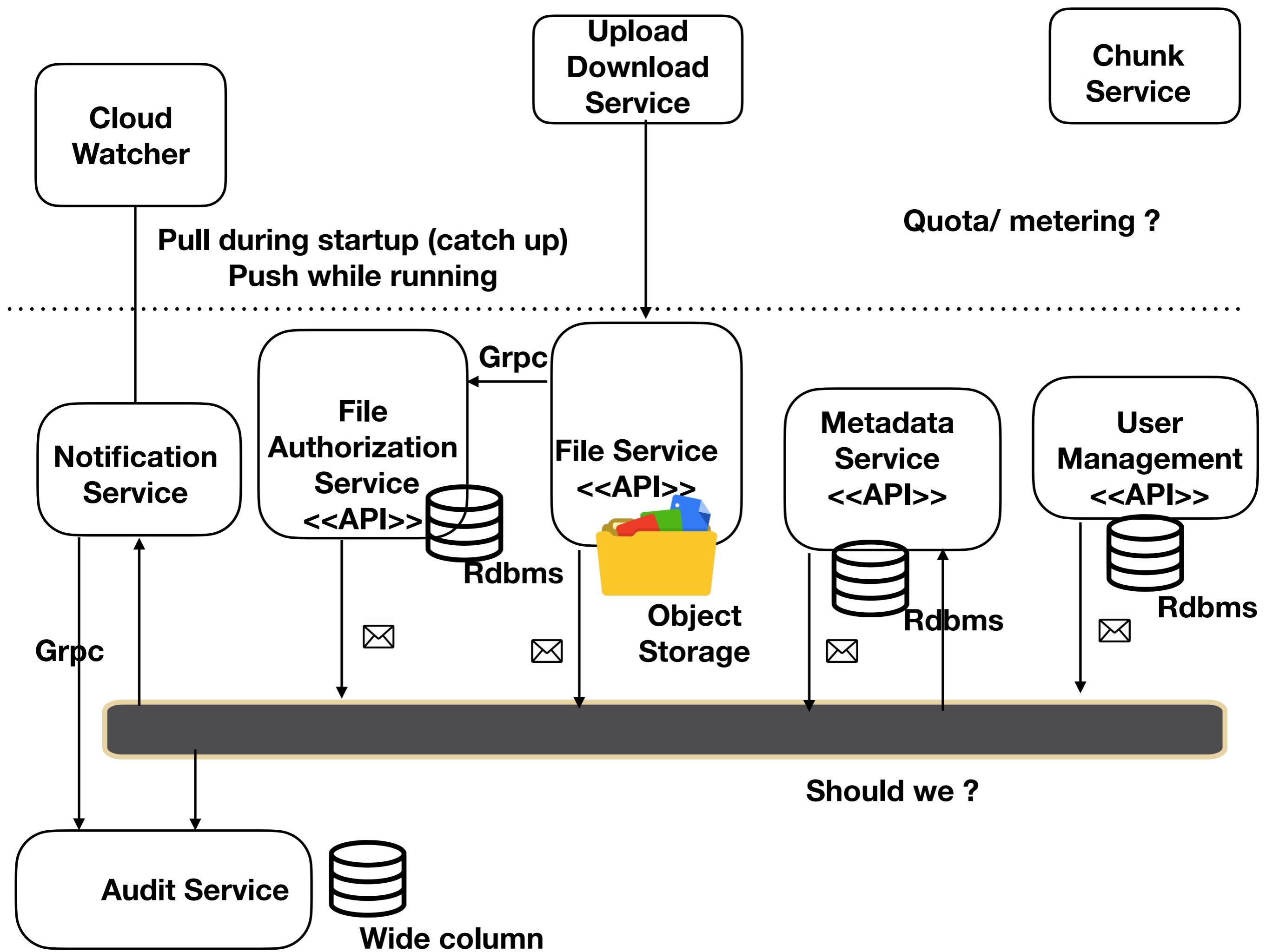


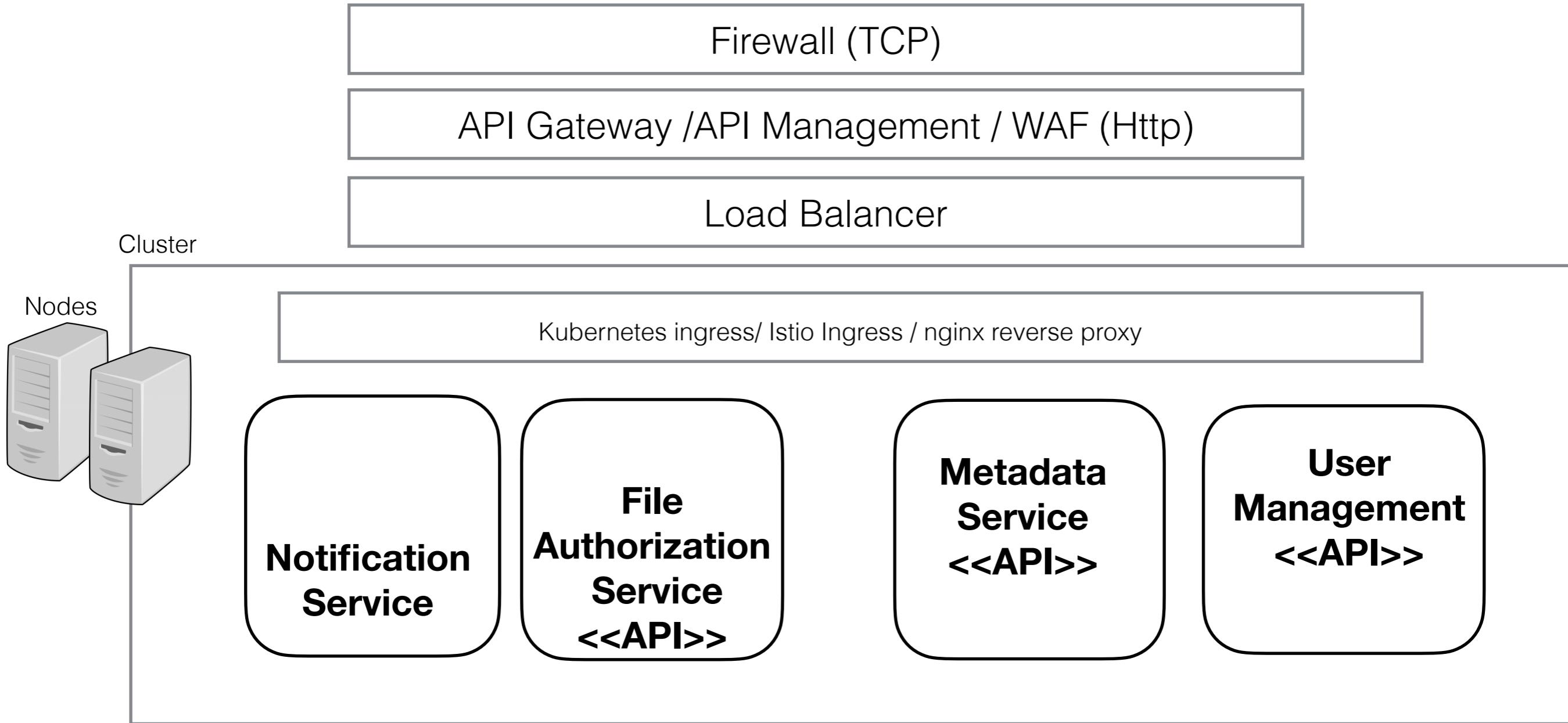
DropBox

Logical View

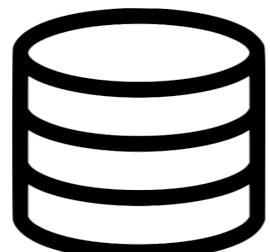








**Object
Storage**



Wide column

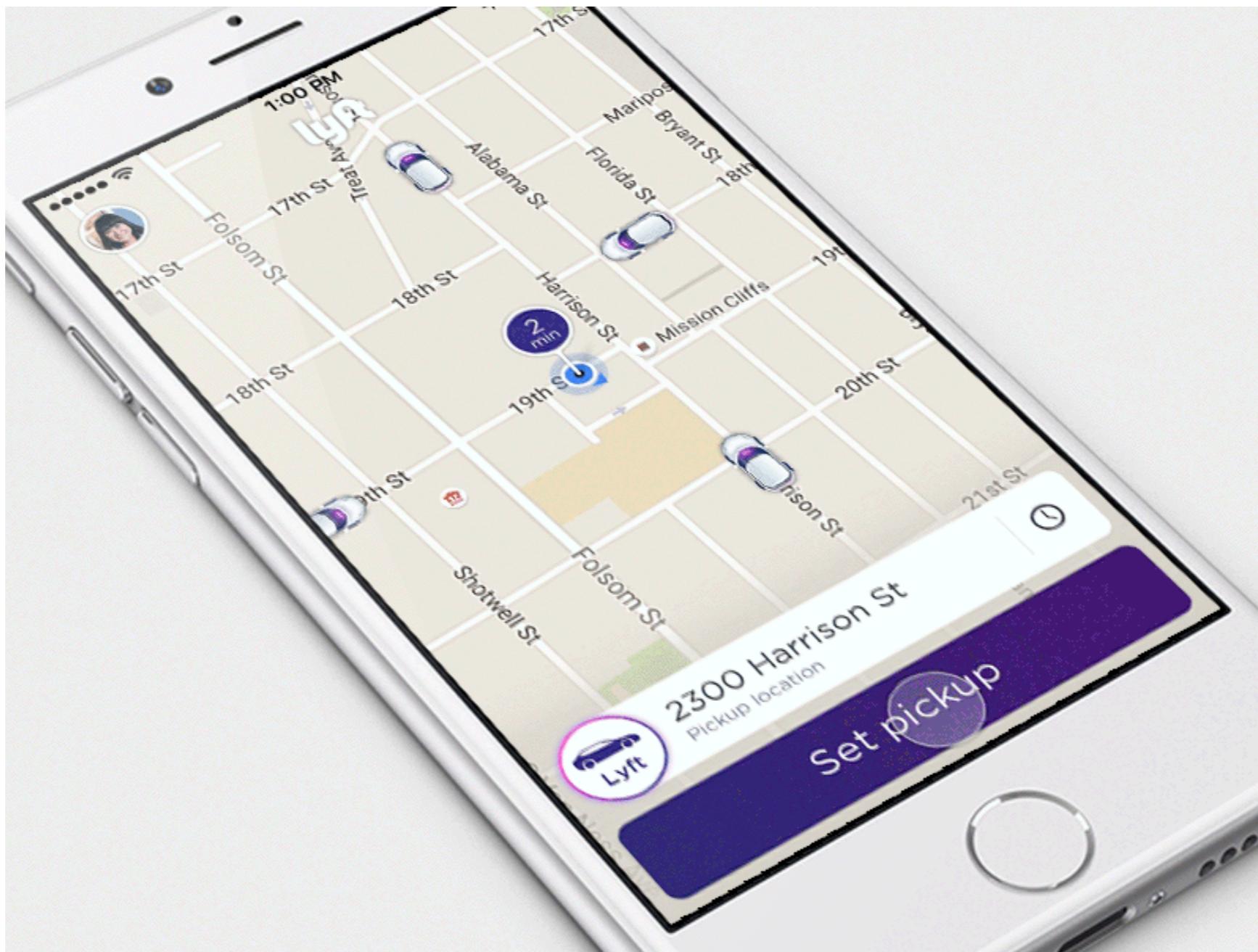


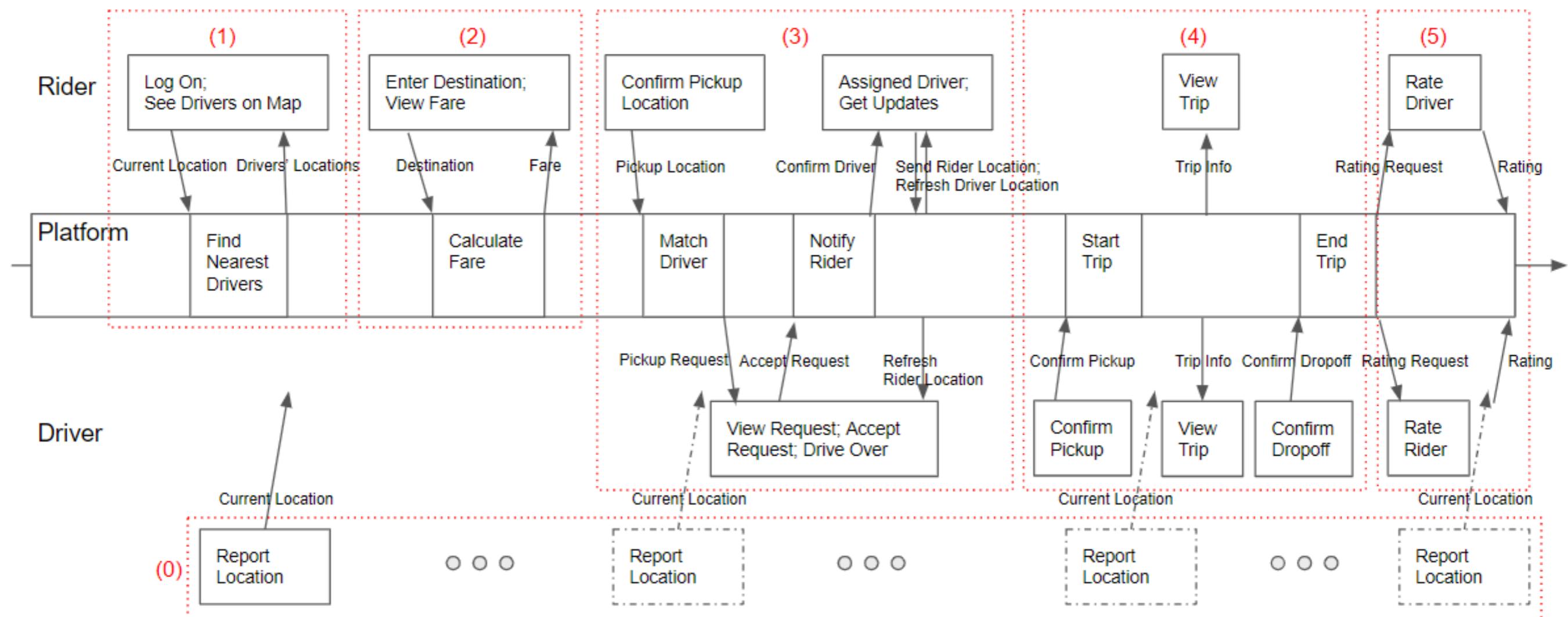
Rdbms



**Message
Queue**

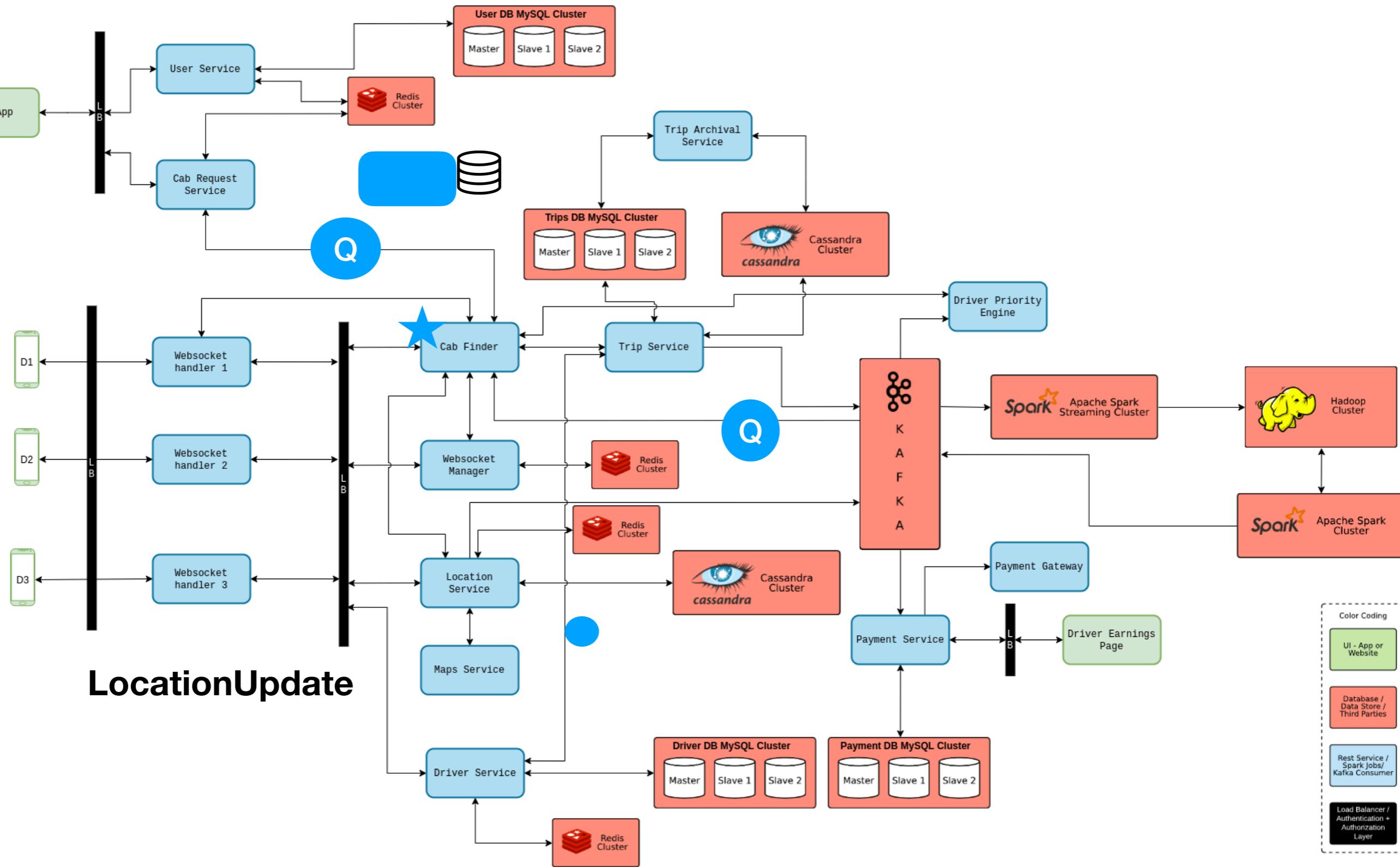
Uber reference architecture

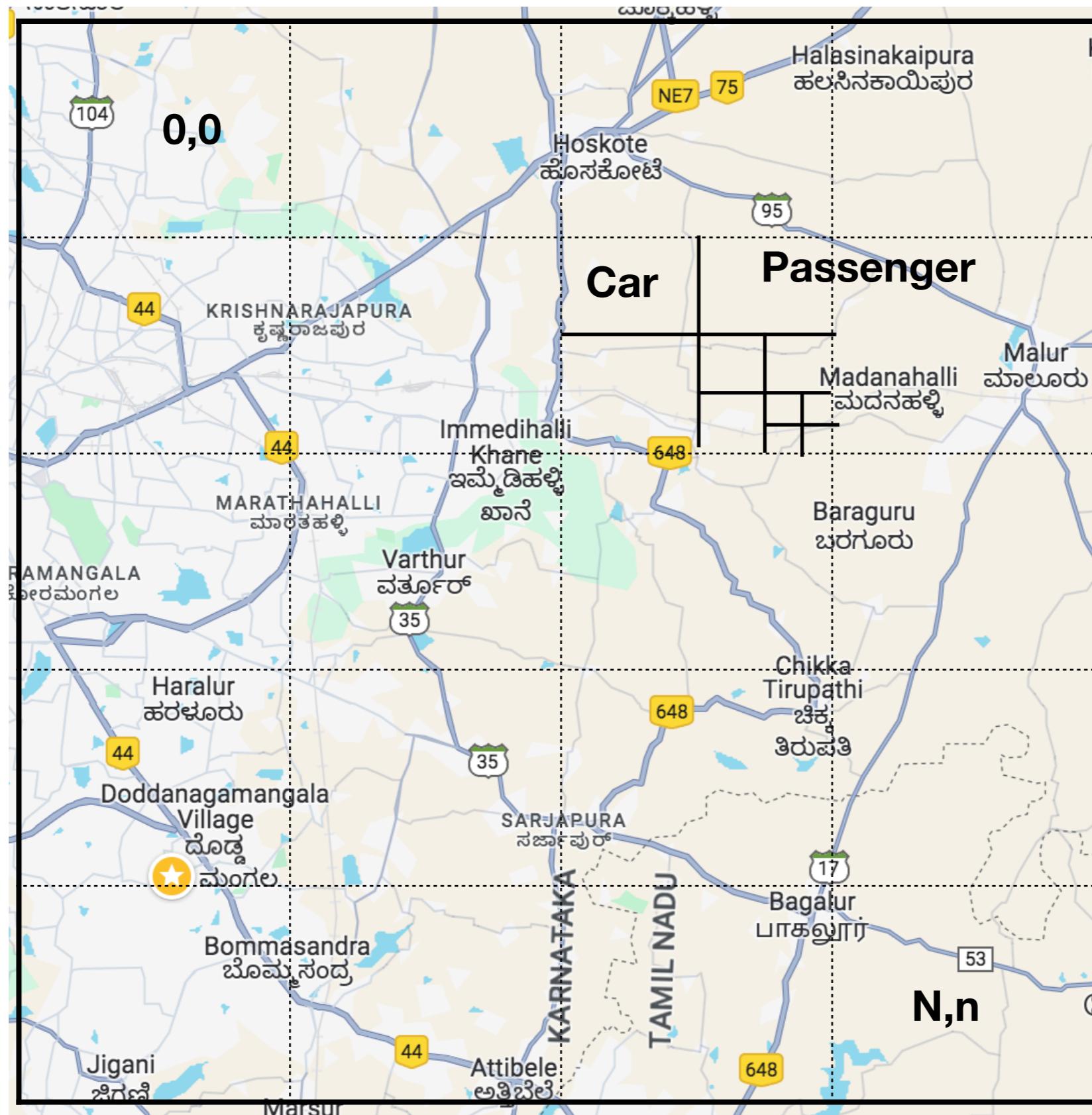




Uber/Lyft/Ola System Design

<code karle>

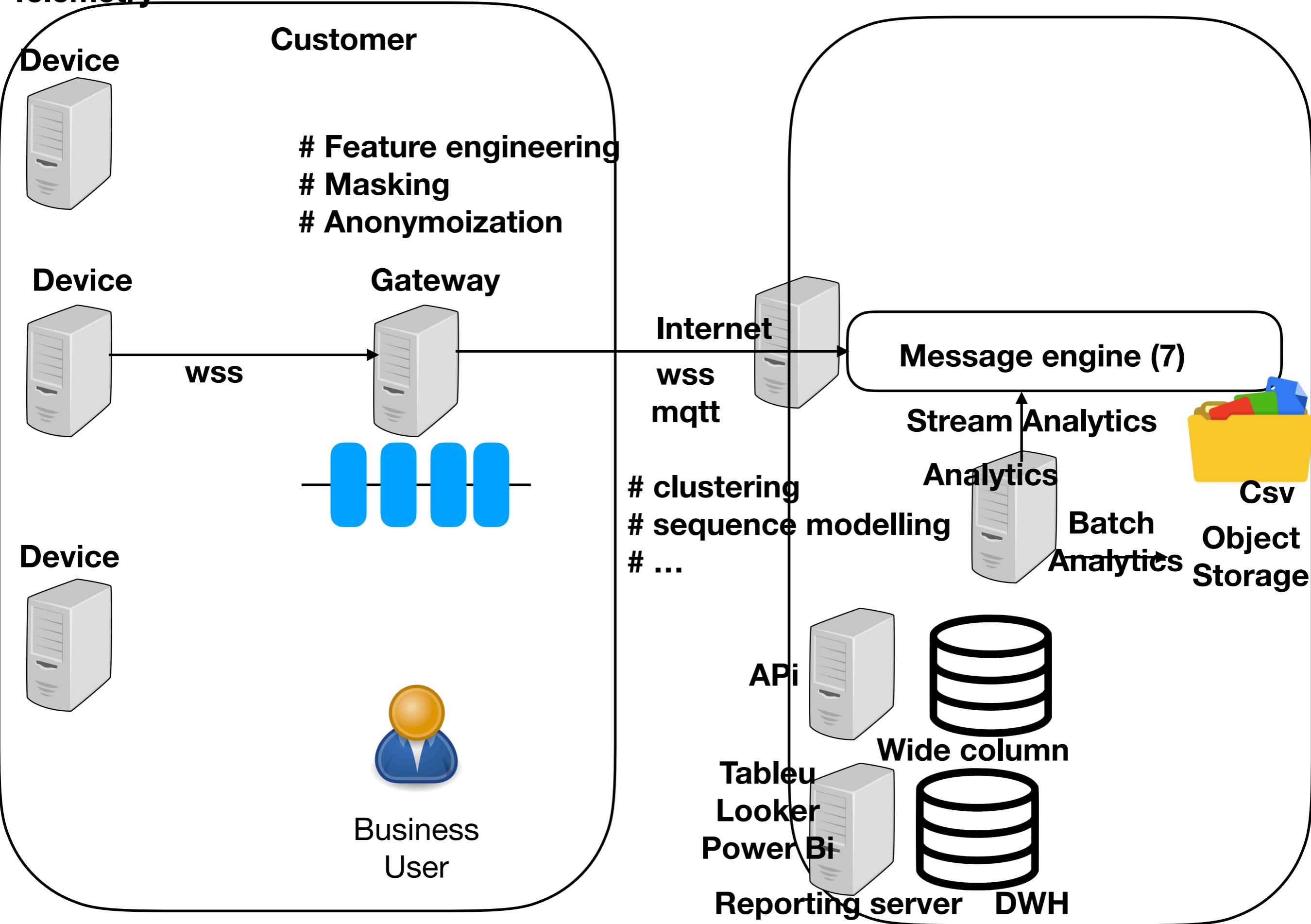




1.collect Telemetry

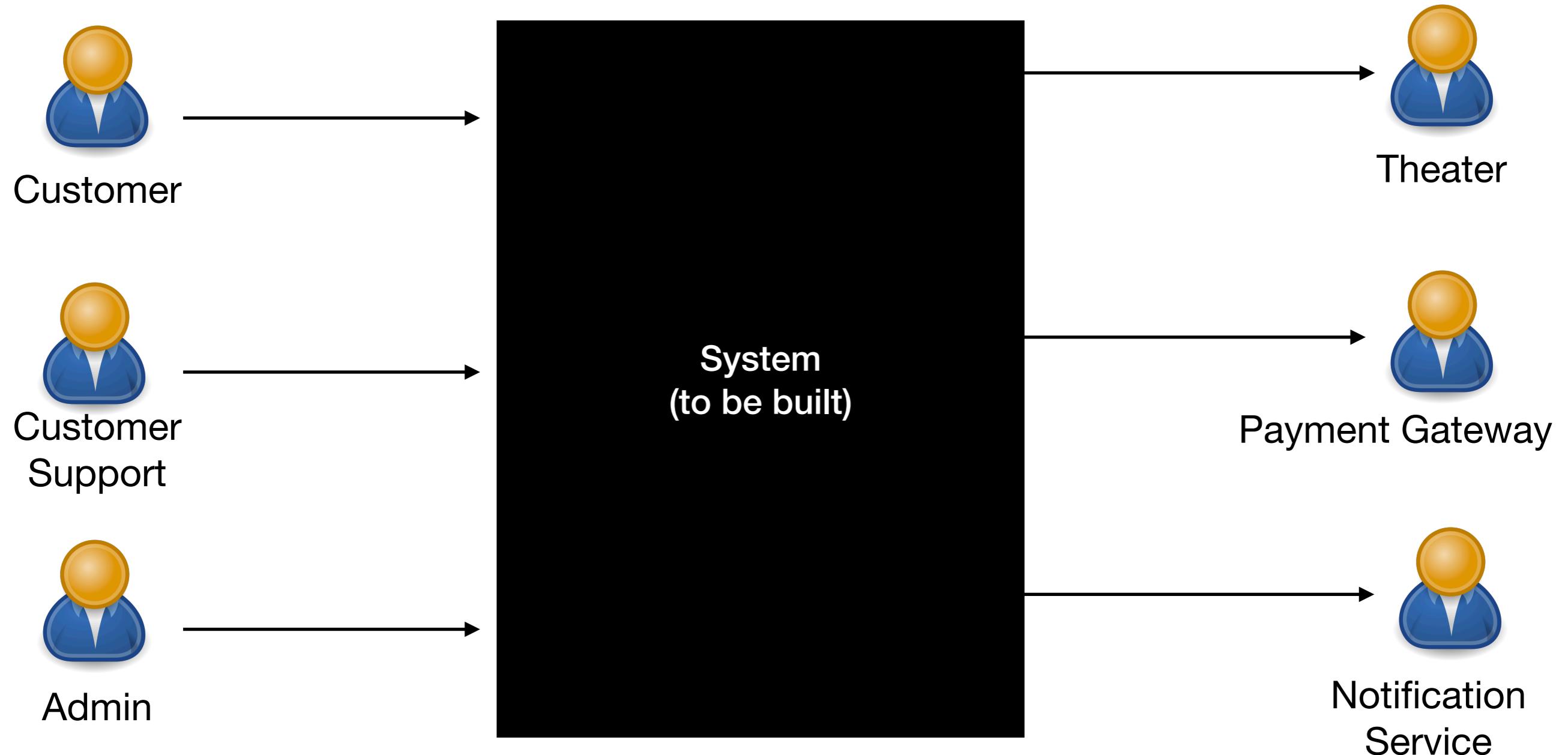
2.secure

3. Analyse

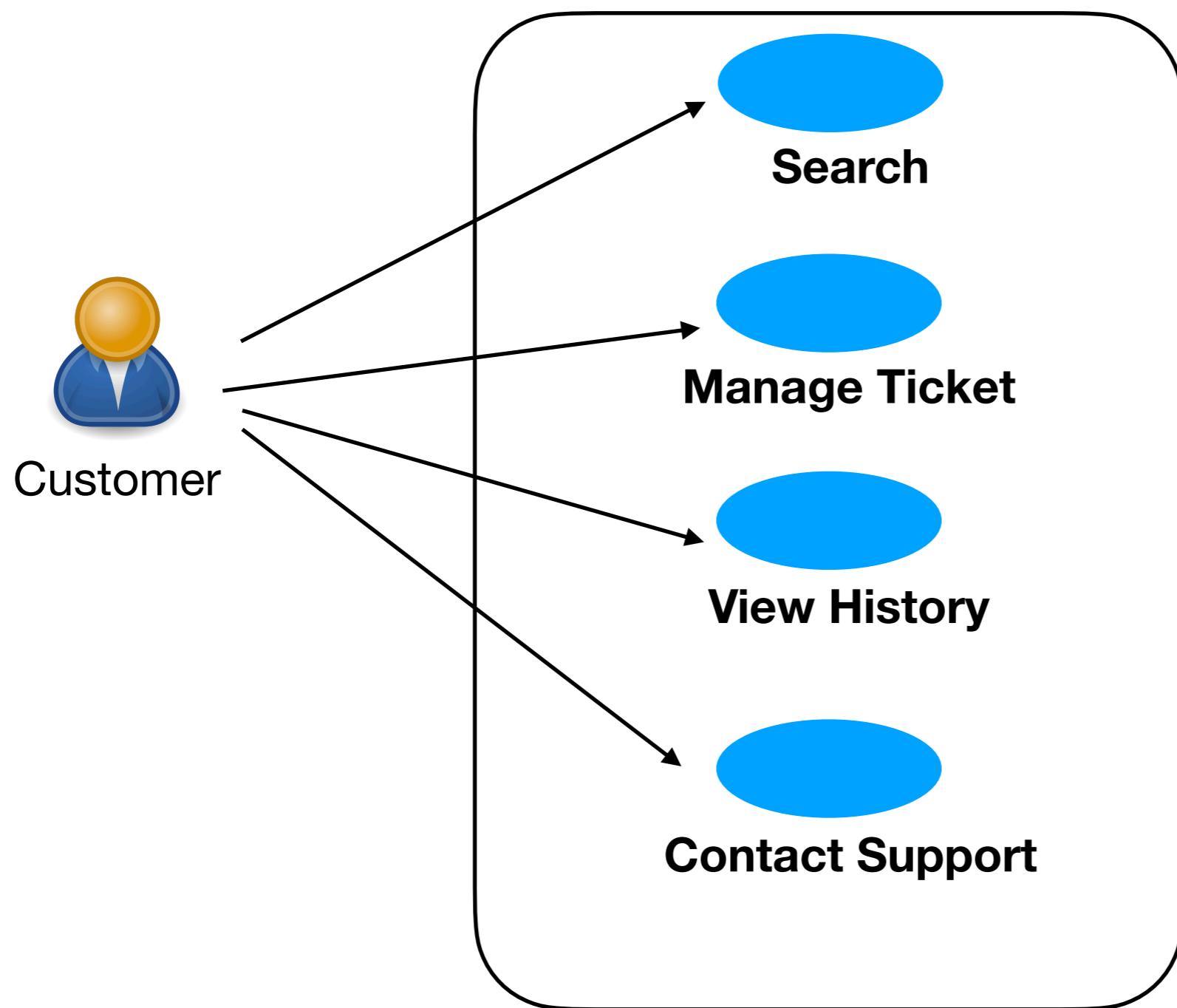


Bookmyshow

Context View



Functional View



Constraints

GDPR Compliance

Less than 50K monthly cost

Should work with old browsers like IE 7.0

Should use AWS cloud .

Should be in production before x

Total deployment budget should not exceed 10000\$ a month

Quality	Source (who)	Stimulus (action)	Artifact (which)	Environment (context)	Response (output)	Measure (scale)
Performance	As a user	I want to book a ticket	Using the Mobile App	During peak load	Ticket is booked	In < 2 seconds
Reliability	As a user	I want to book a ticket	Using the Web Portal App	receives a duplicate request	User is not double-charged	In 100% of the cases
Availability	Internal (unspecified)	Failure	Cluster	Normal operating environment	Detect a fault, perform provide failover, and recover fully.	Detect a fault within 1 second, provide failover within 15 seconds, and fully recover from failure within 2 minutes
Security	External to system	flood of calls	through the Web Service endpoint	Normal operation	notifies system administrators	continues to service requests in degraded mode

Assumption

Will use open source

During peak load, 25k users will be booking tickets

Logical View

System



Booking

Search

Users

Payment

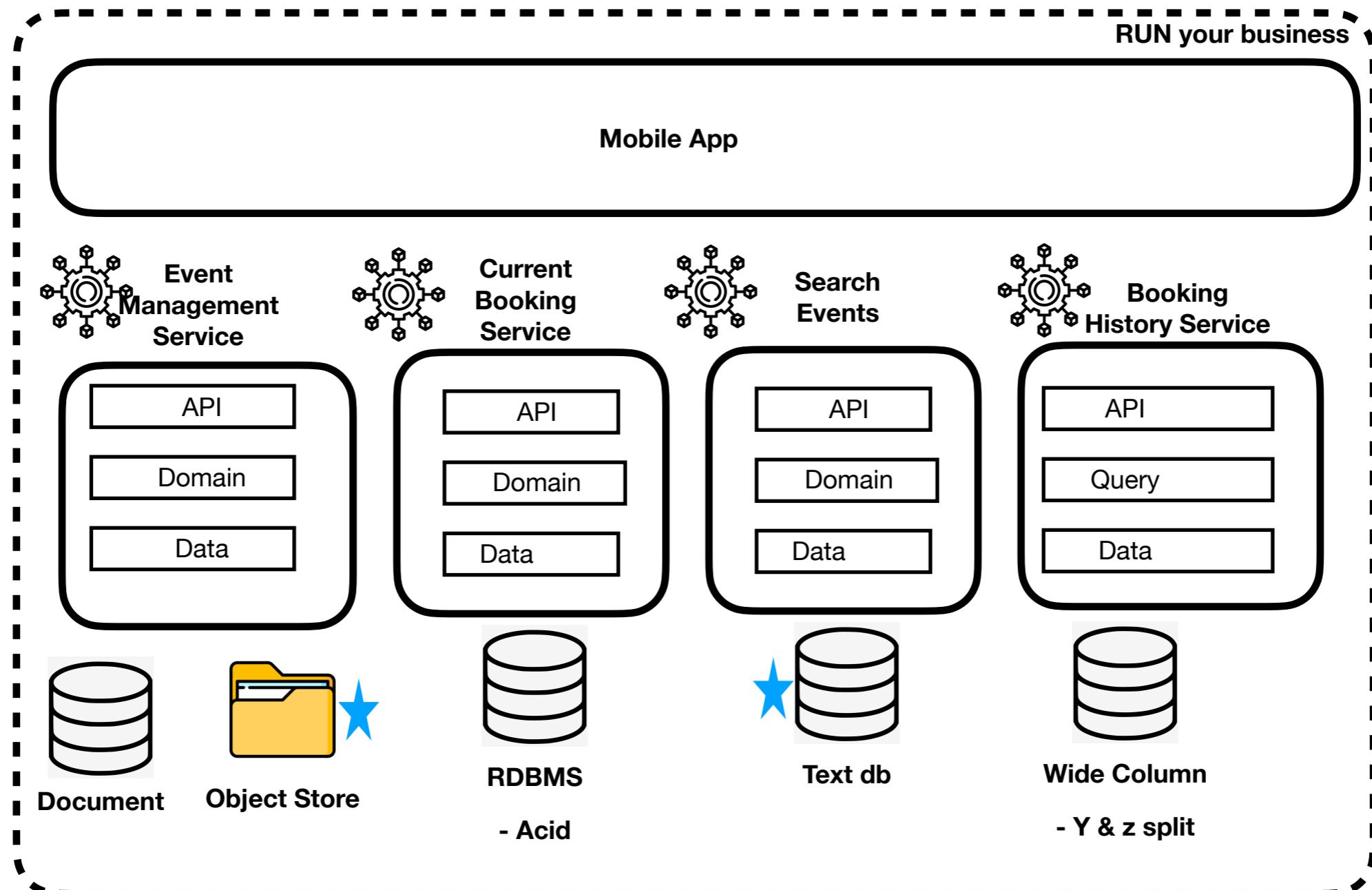
Notification

Inventory

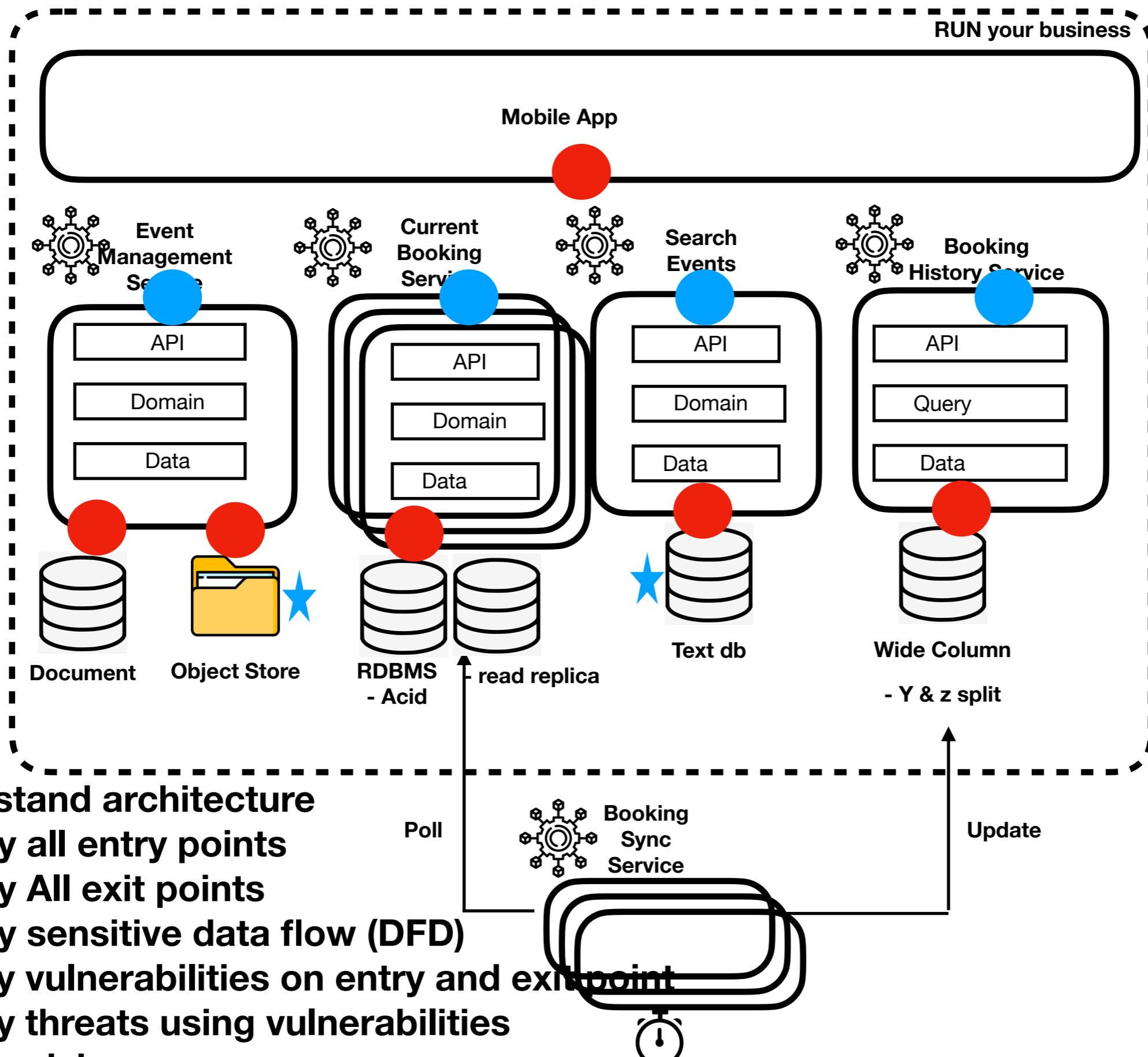
Reviews

Insights

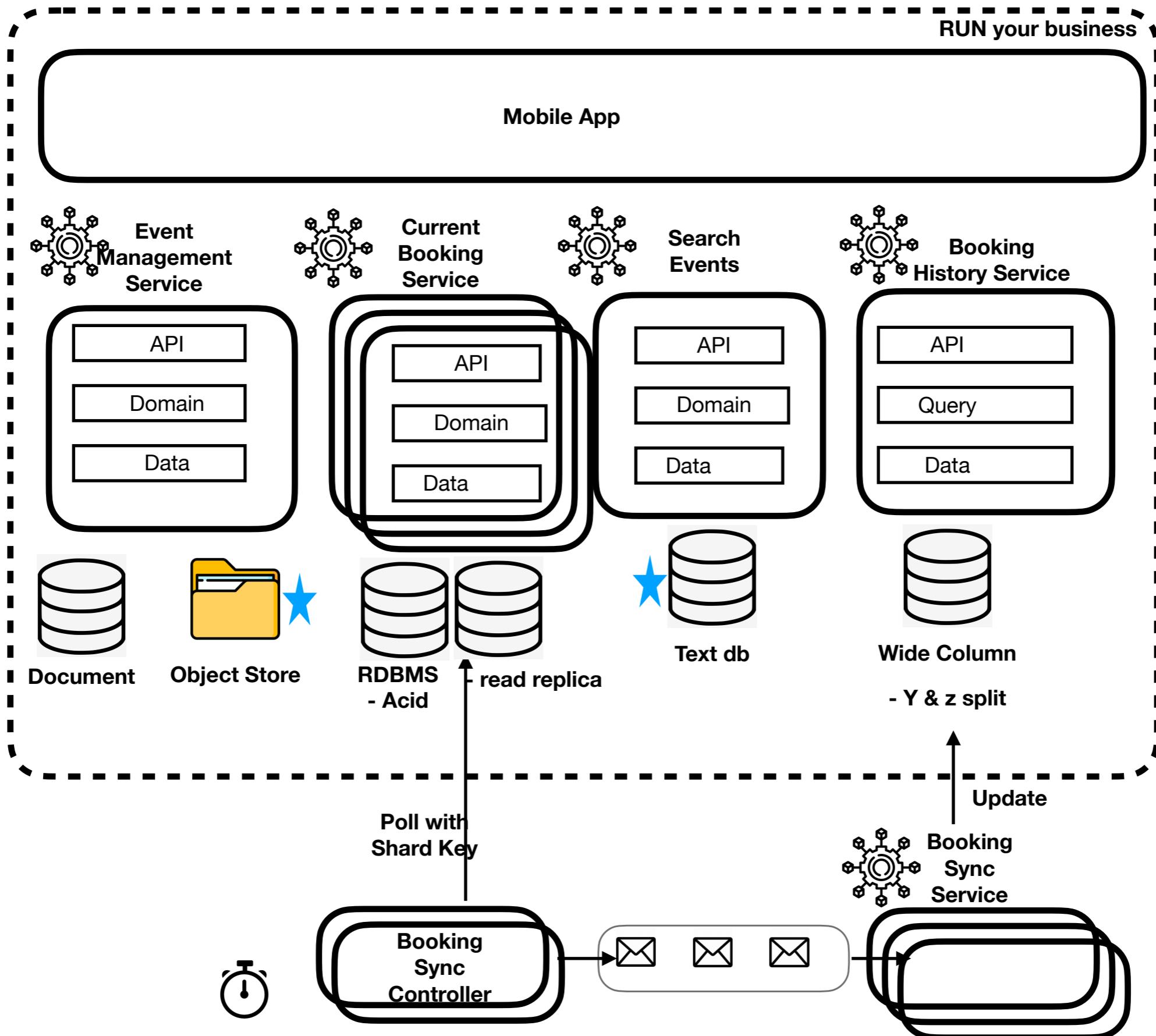
Operation Architecture



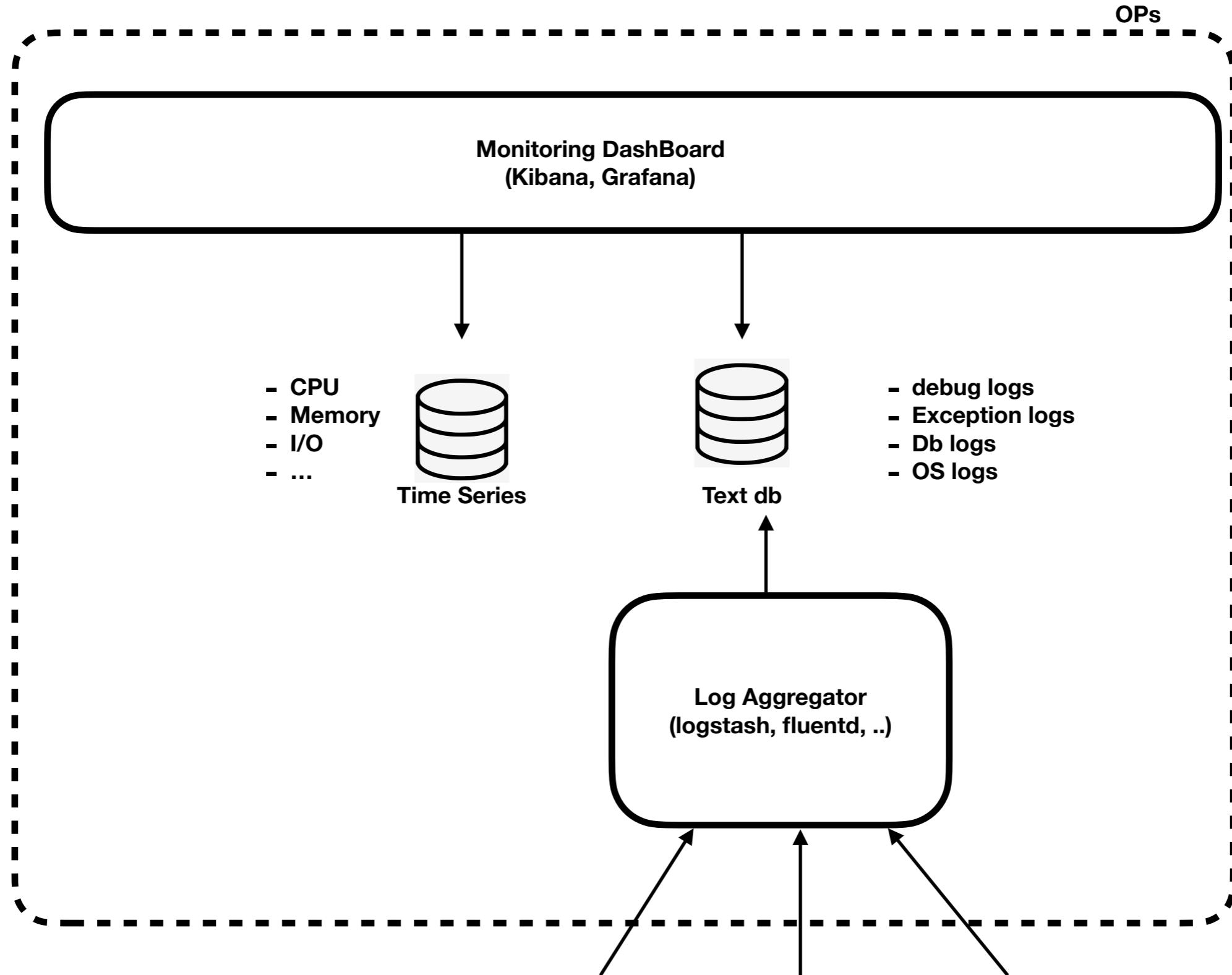
Operation Architecture



Operation Architecture

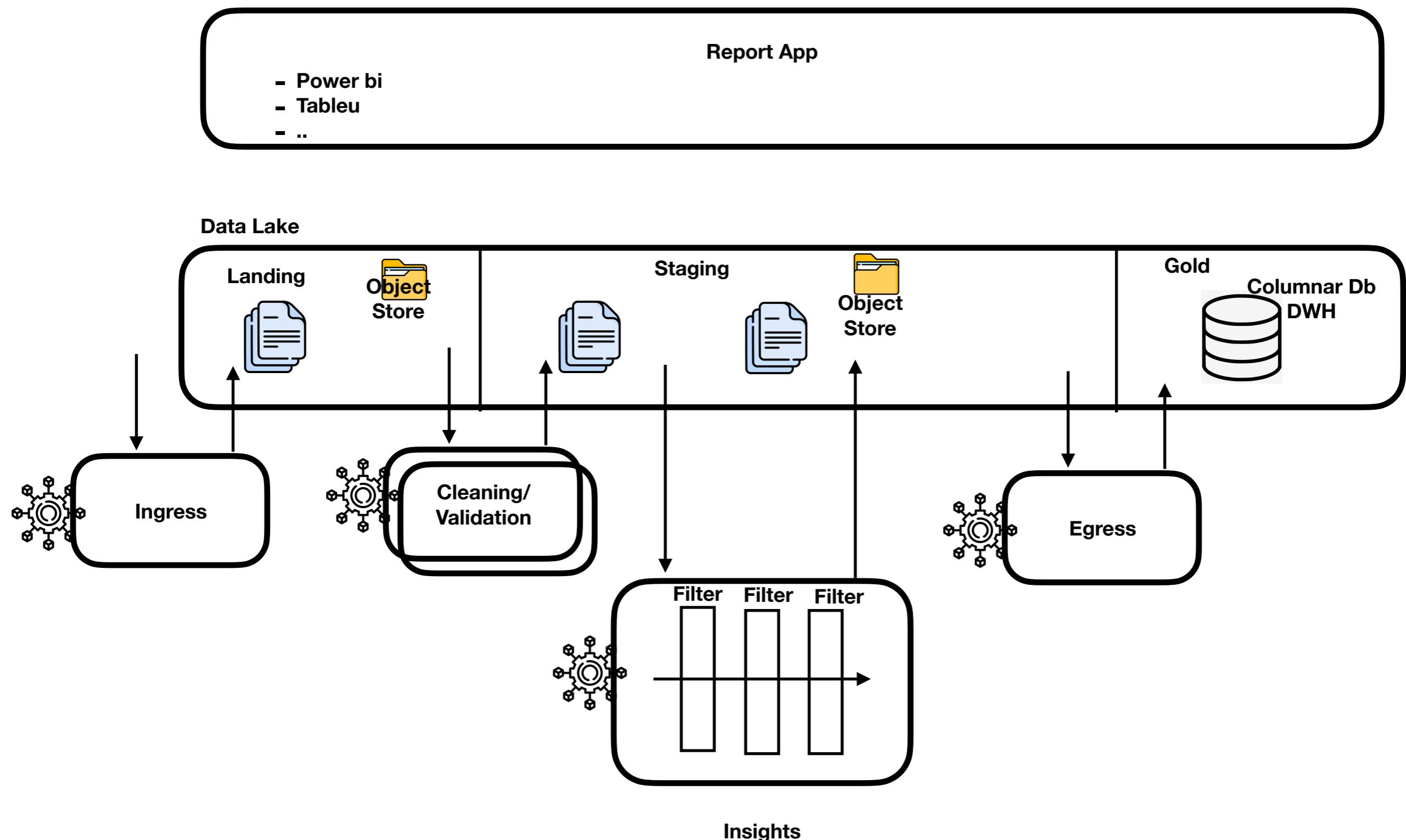


Monitoring Architecture

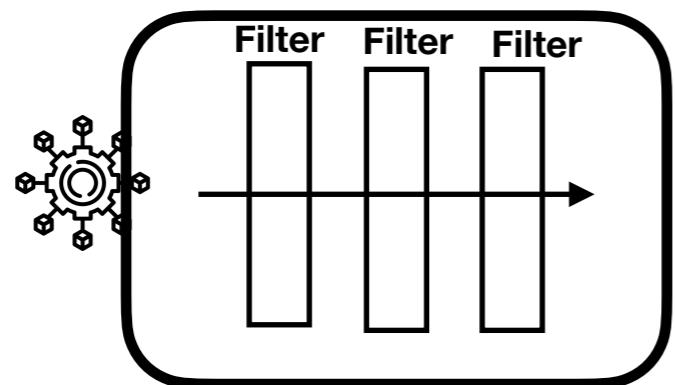


Analytical Architecture

Manage your business

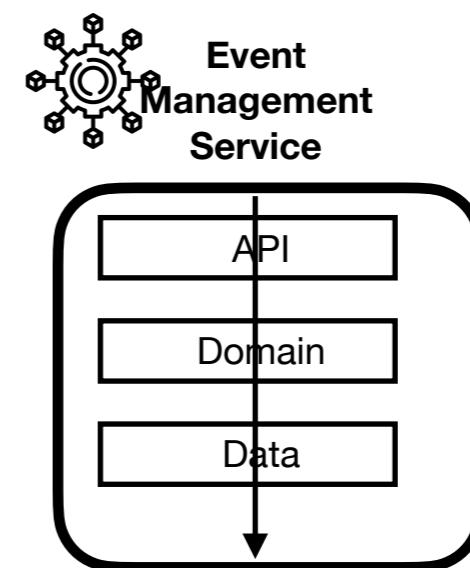


Separation of Data transformation based on functional responsibility

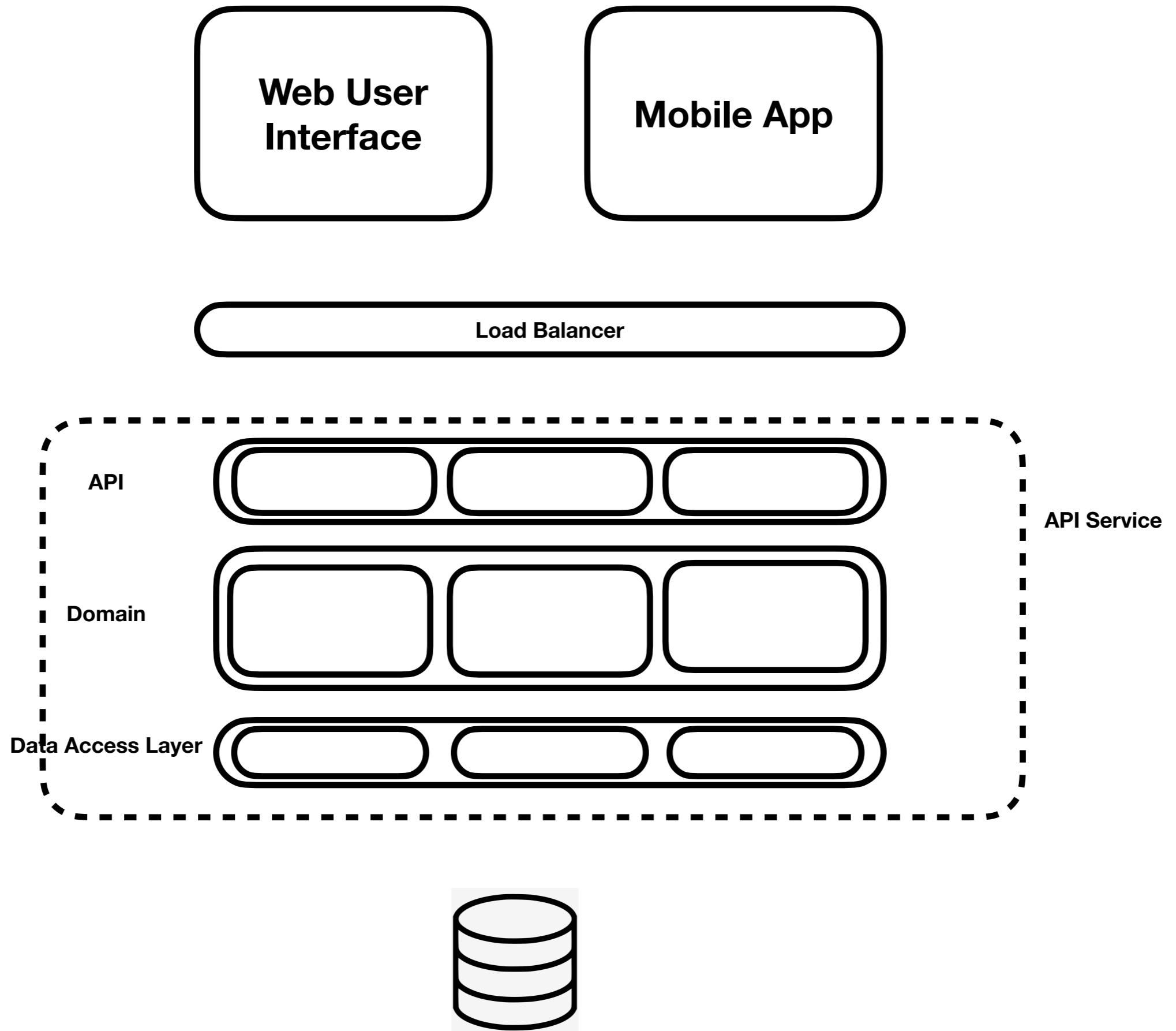


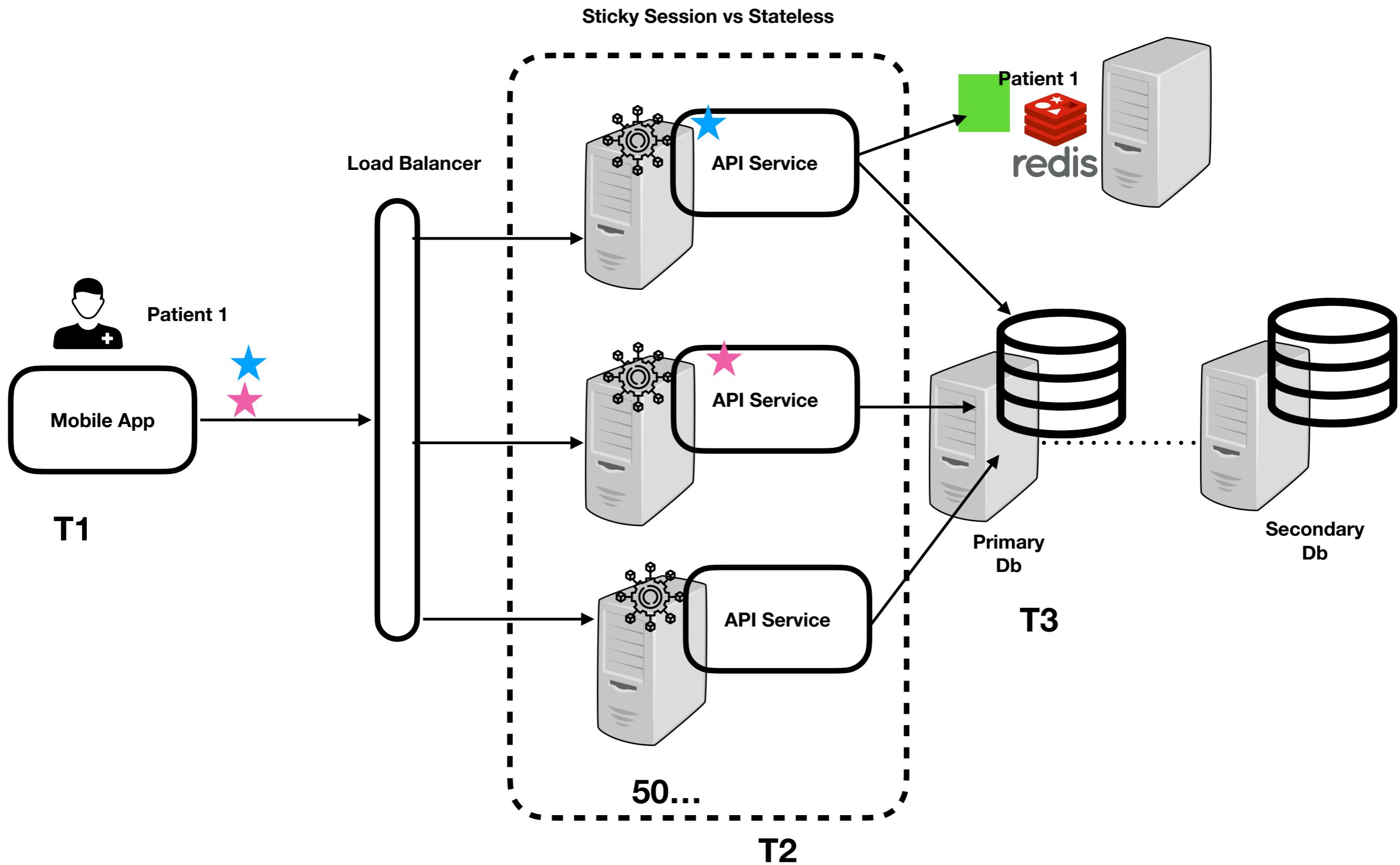
All filters will have same input and output structure

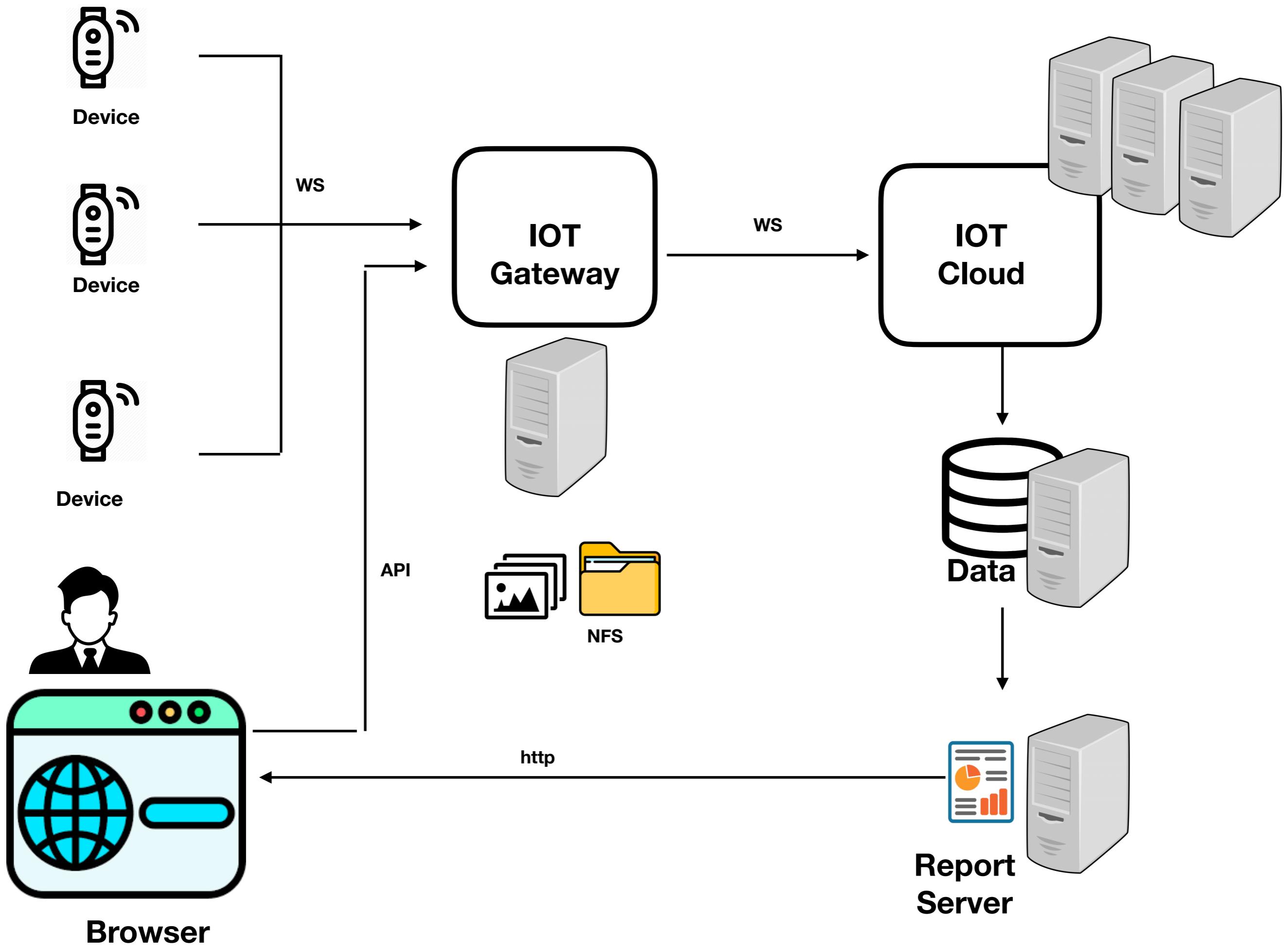
Separation of Data transformation at different technology layers

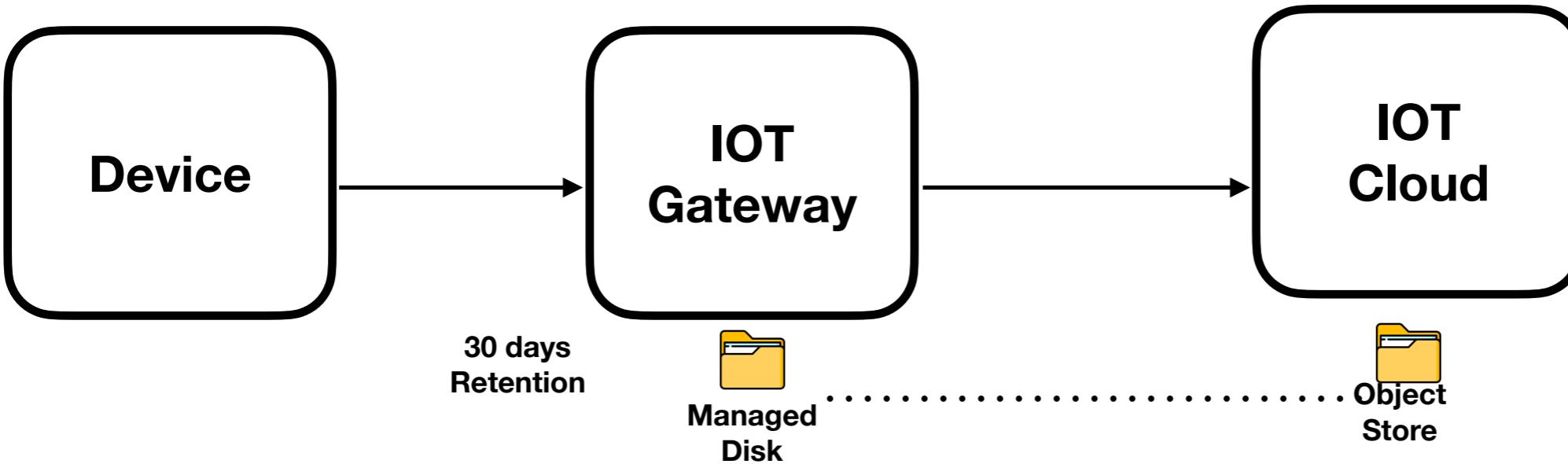
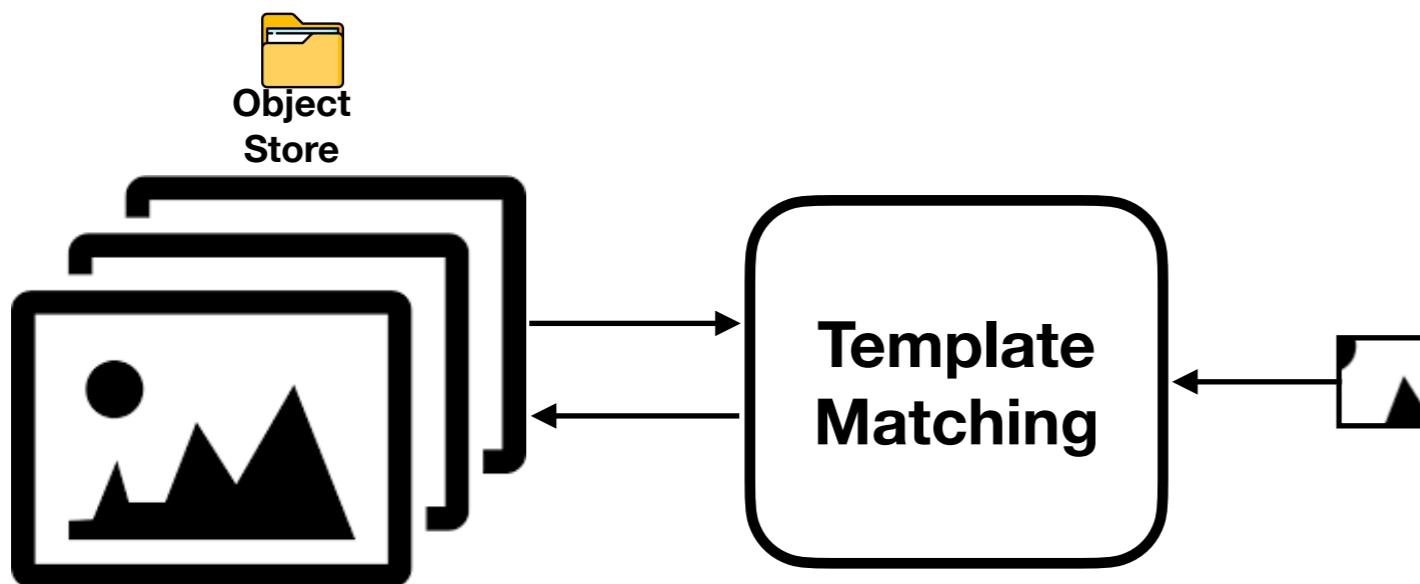


Every layer will have different input and output structure

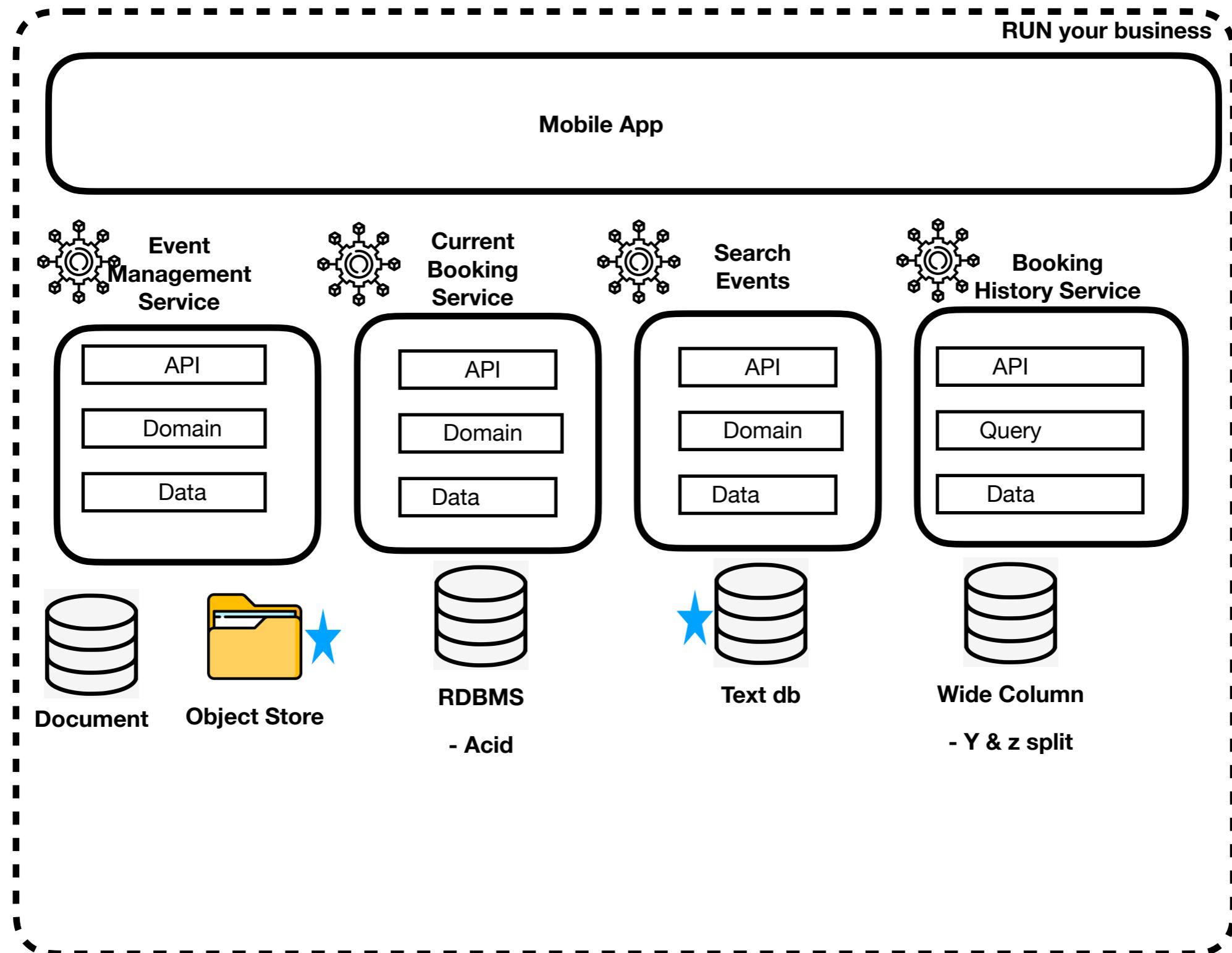






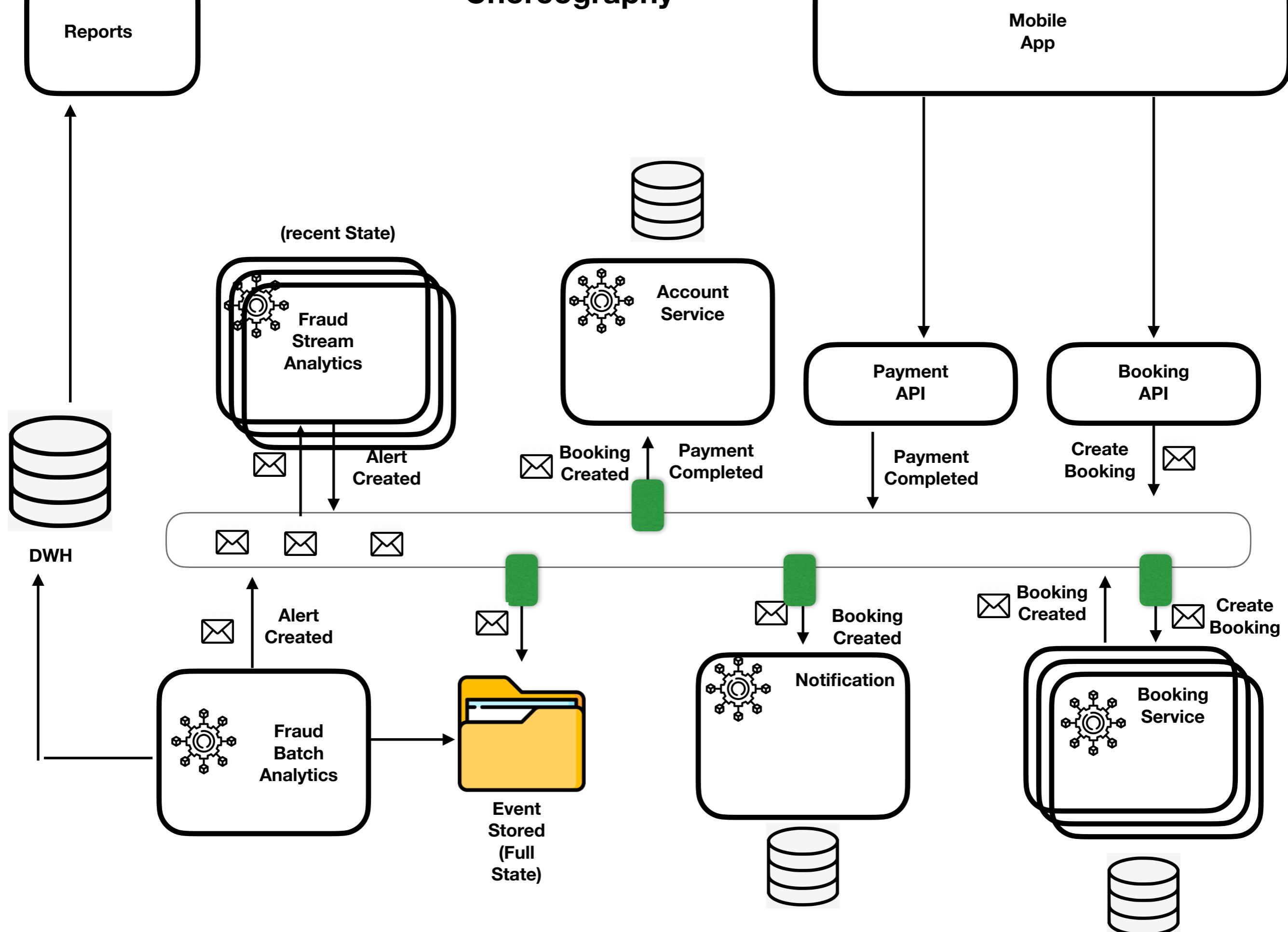


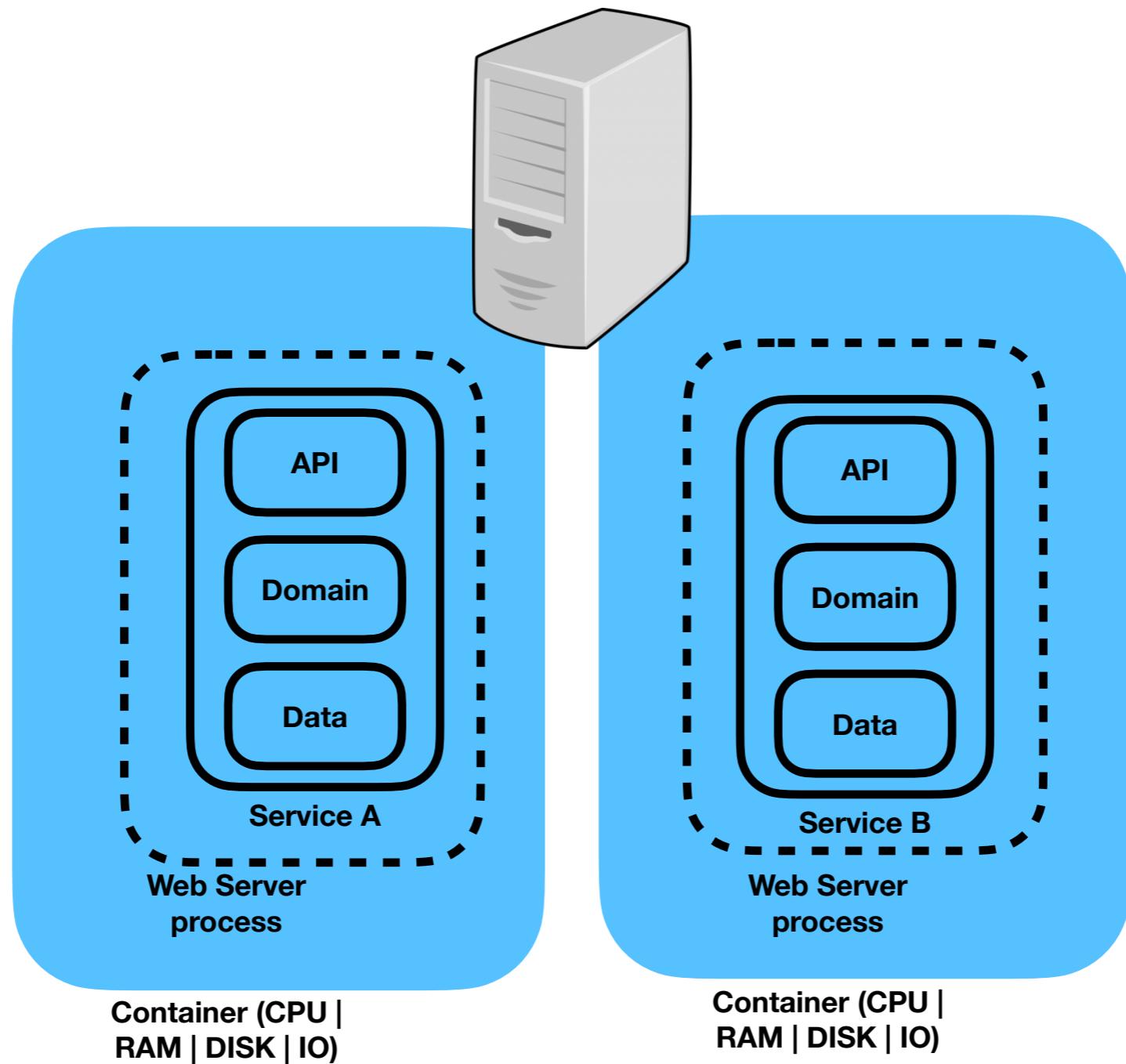
Operation Architecture

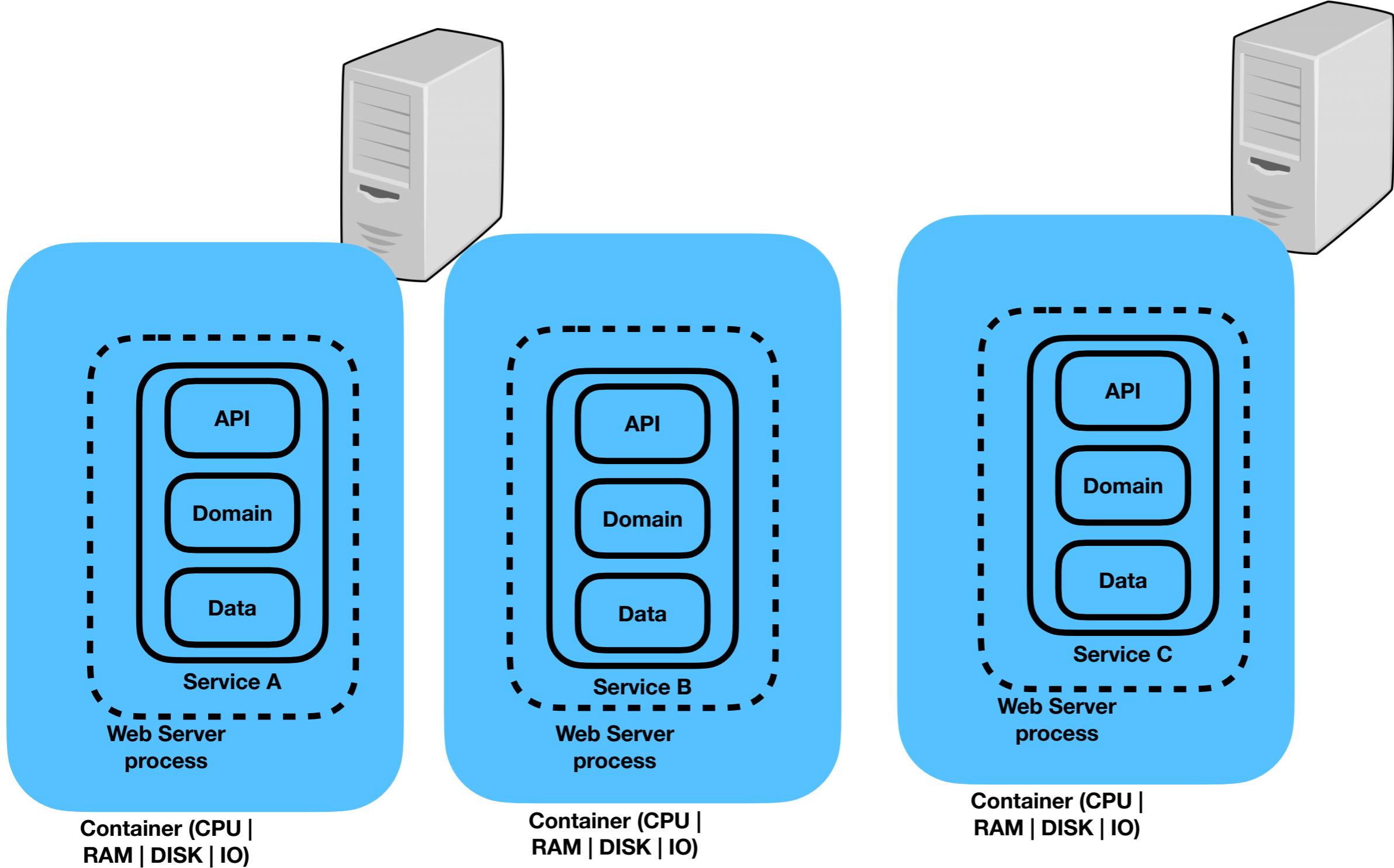


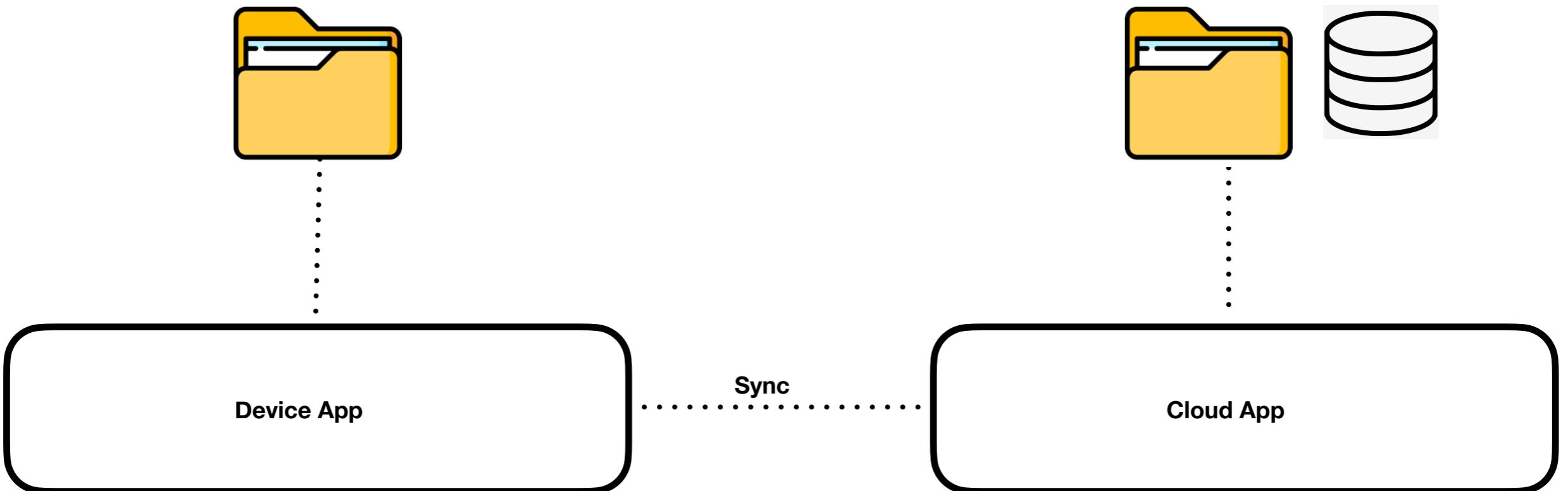
- Responsiveness
 - circuit breaker ?, Throttling, Rate limit
 - Caching
 - ...
- Resilient
 - Cube (split) ?, Cube (Shard) ?, Retry
- Elastic (Scale out / horizontal scale)
 - K8s, Cube, CQRS, Microservice,
- Message driven
 - EDA, Event storming, event sourcing, choreography, ...

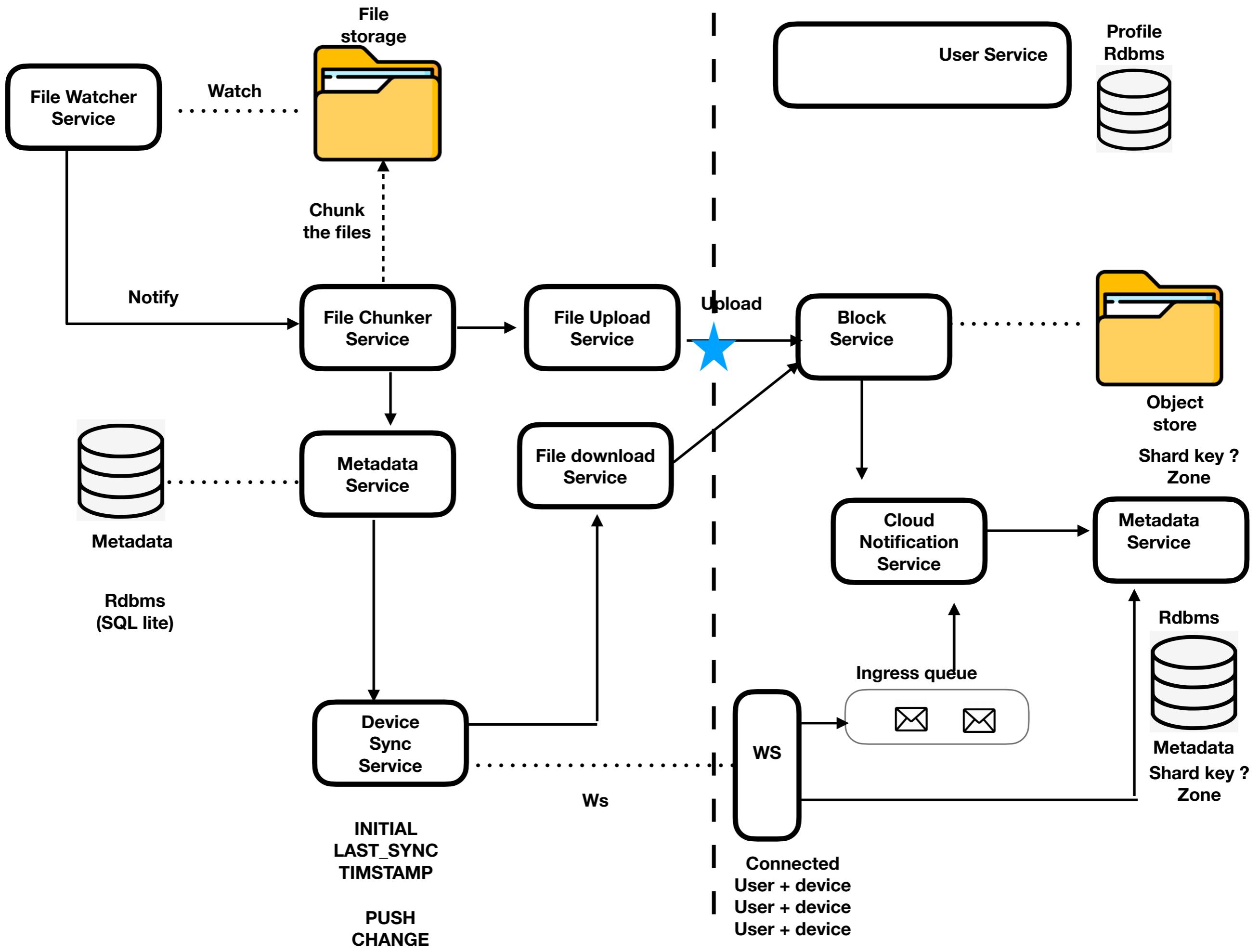
Choreography

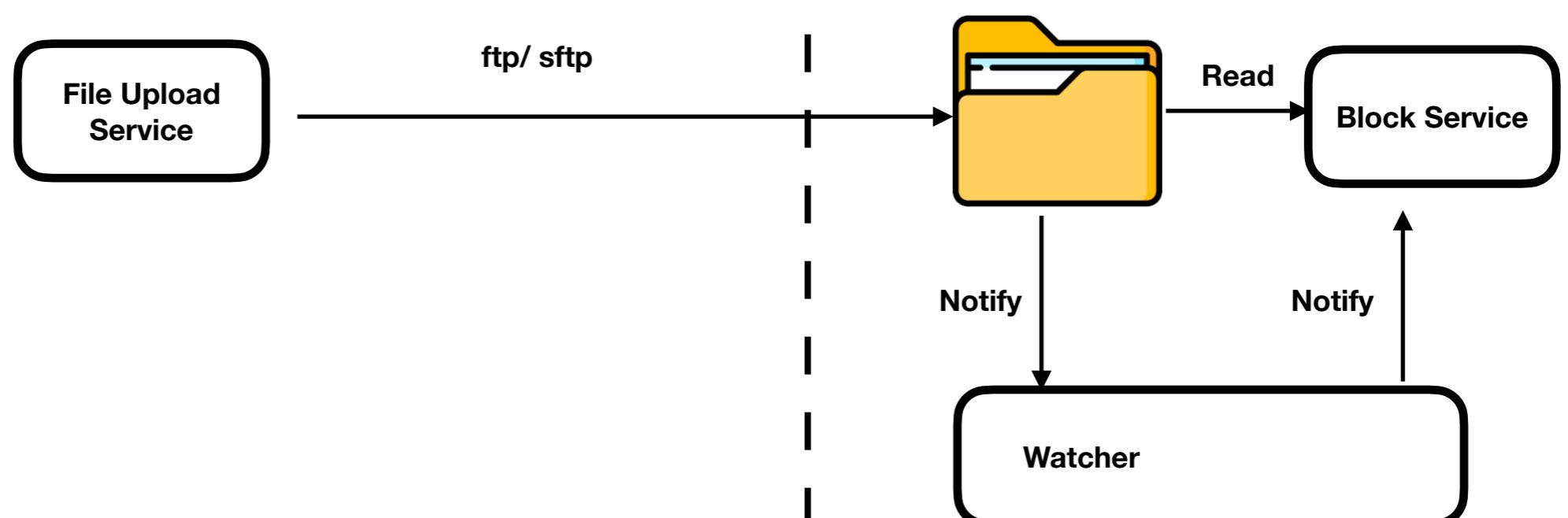






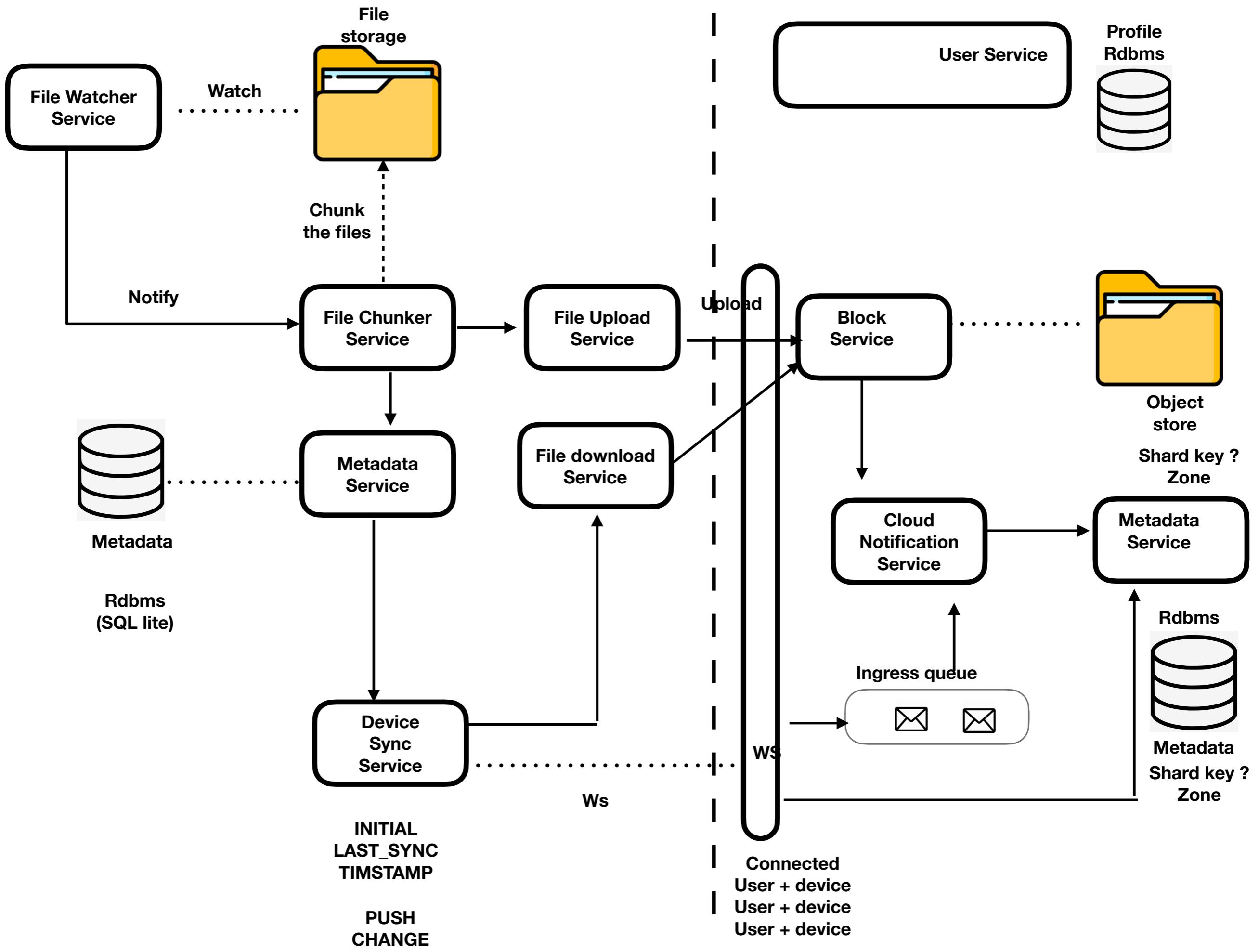


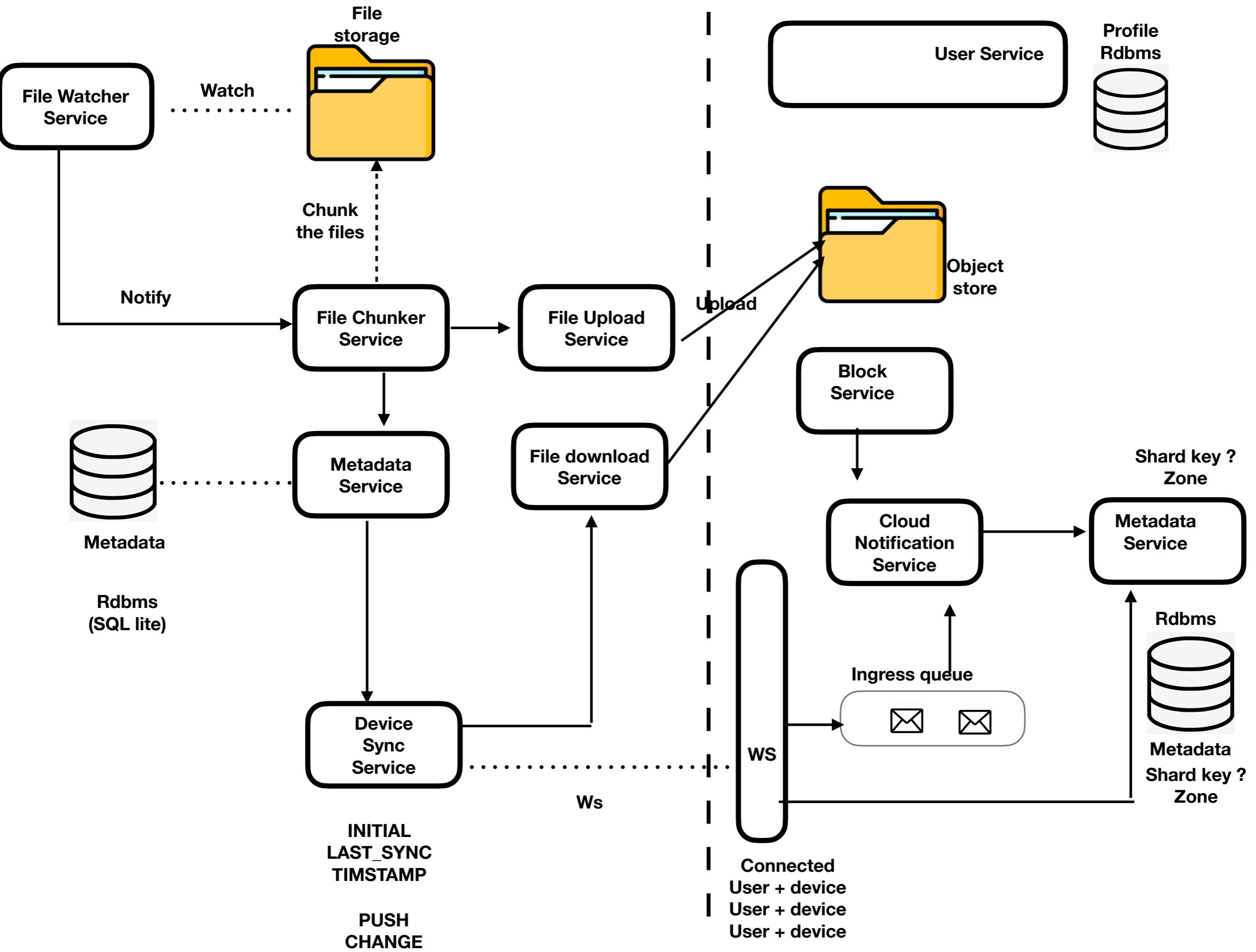




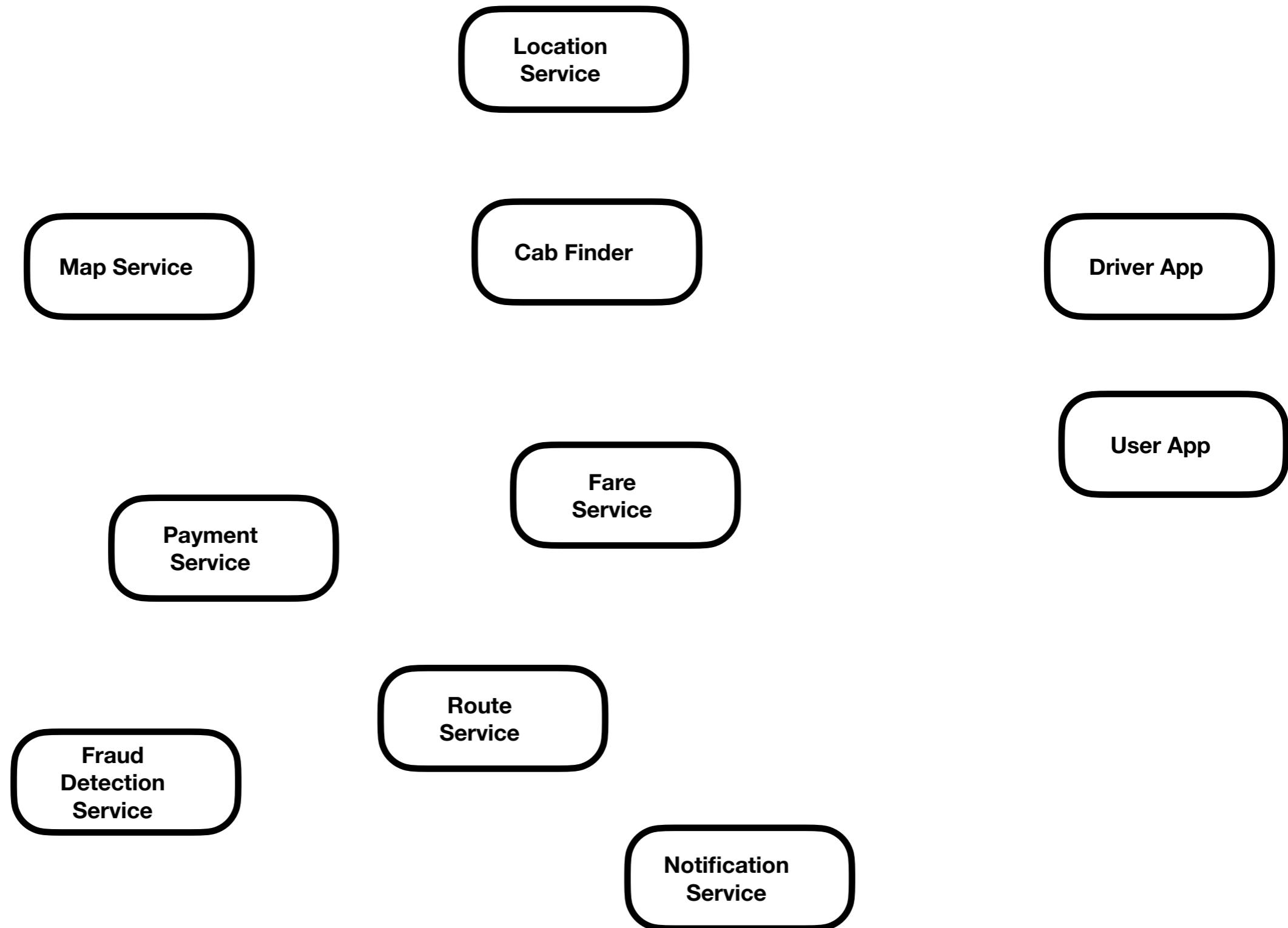
Firewall should be configured ?
Security ?

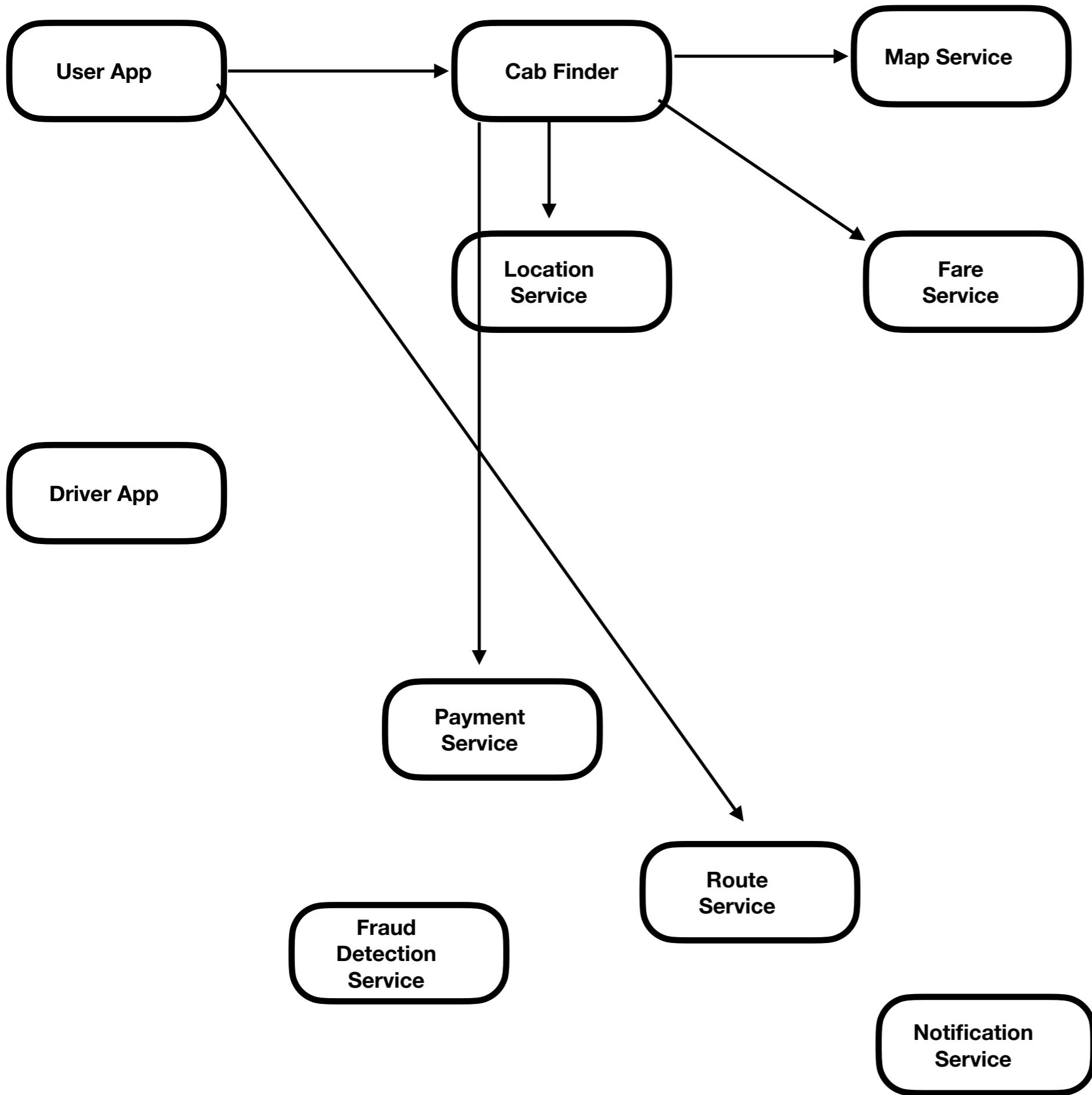


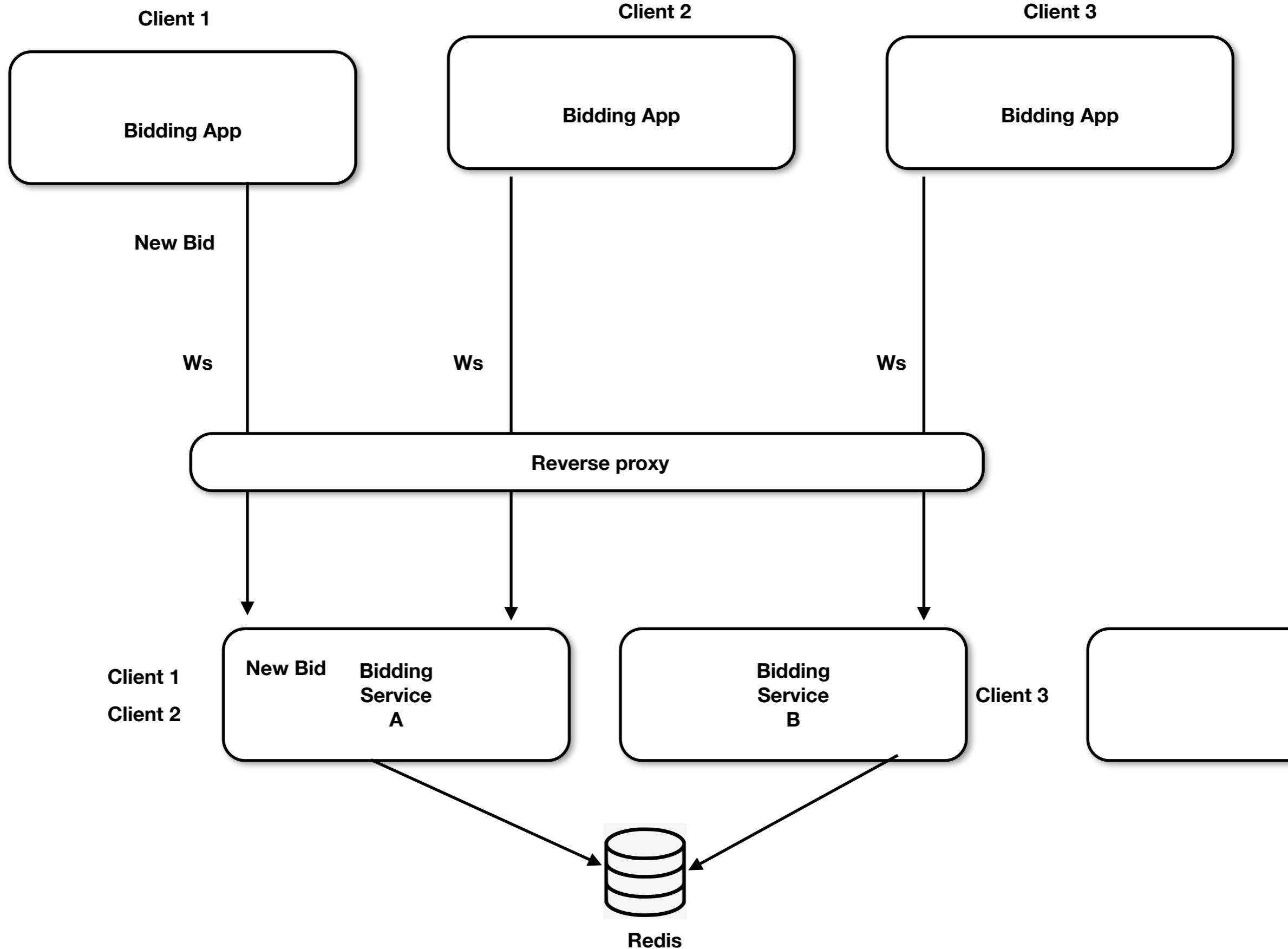


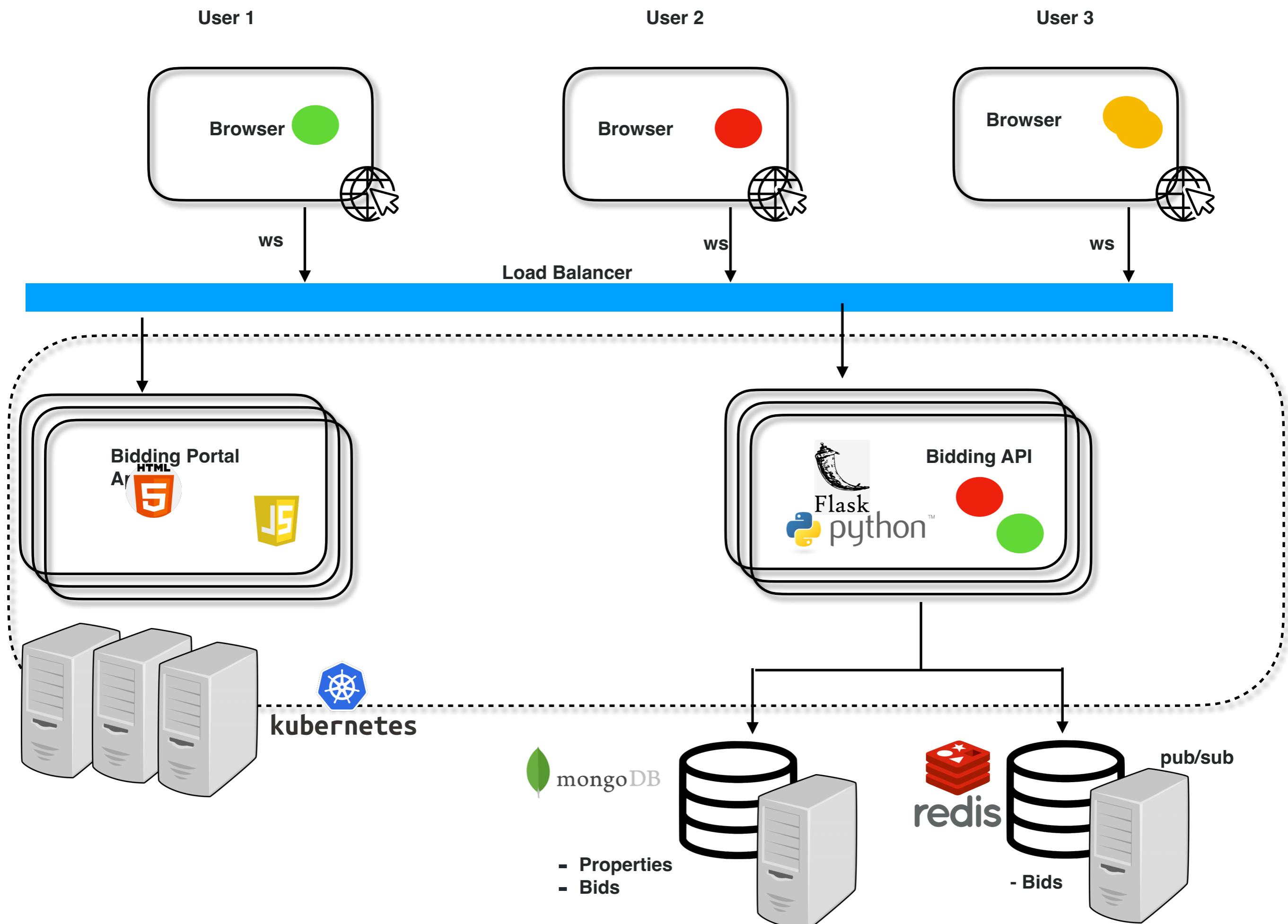


- Multiple/Flexi schema (no)
- Distributed (no)
- Hierarchy structure (tree / xml/ json) (no)
- Indexing on secondary
- Distributed (no)
- Single value
- No secondary index



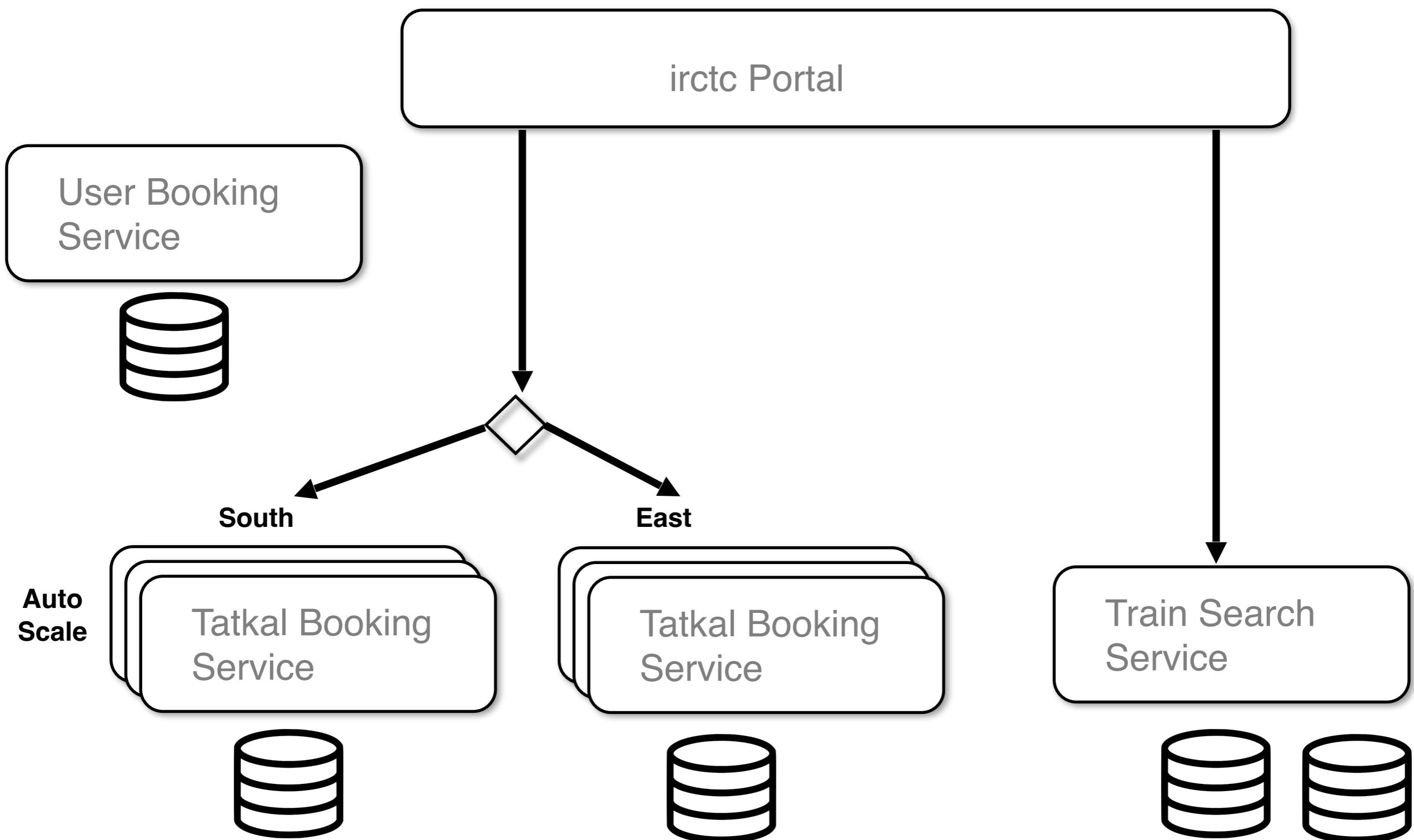


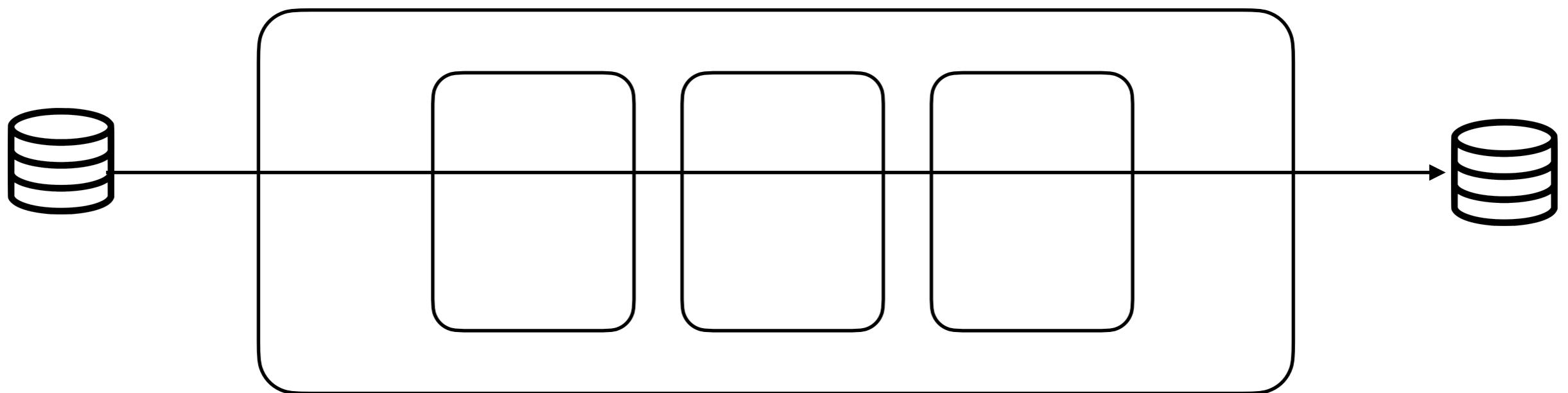
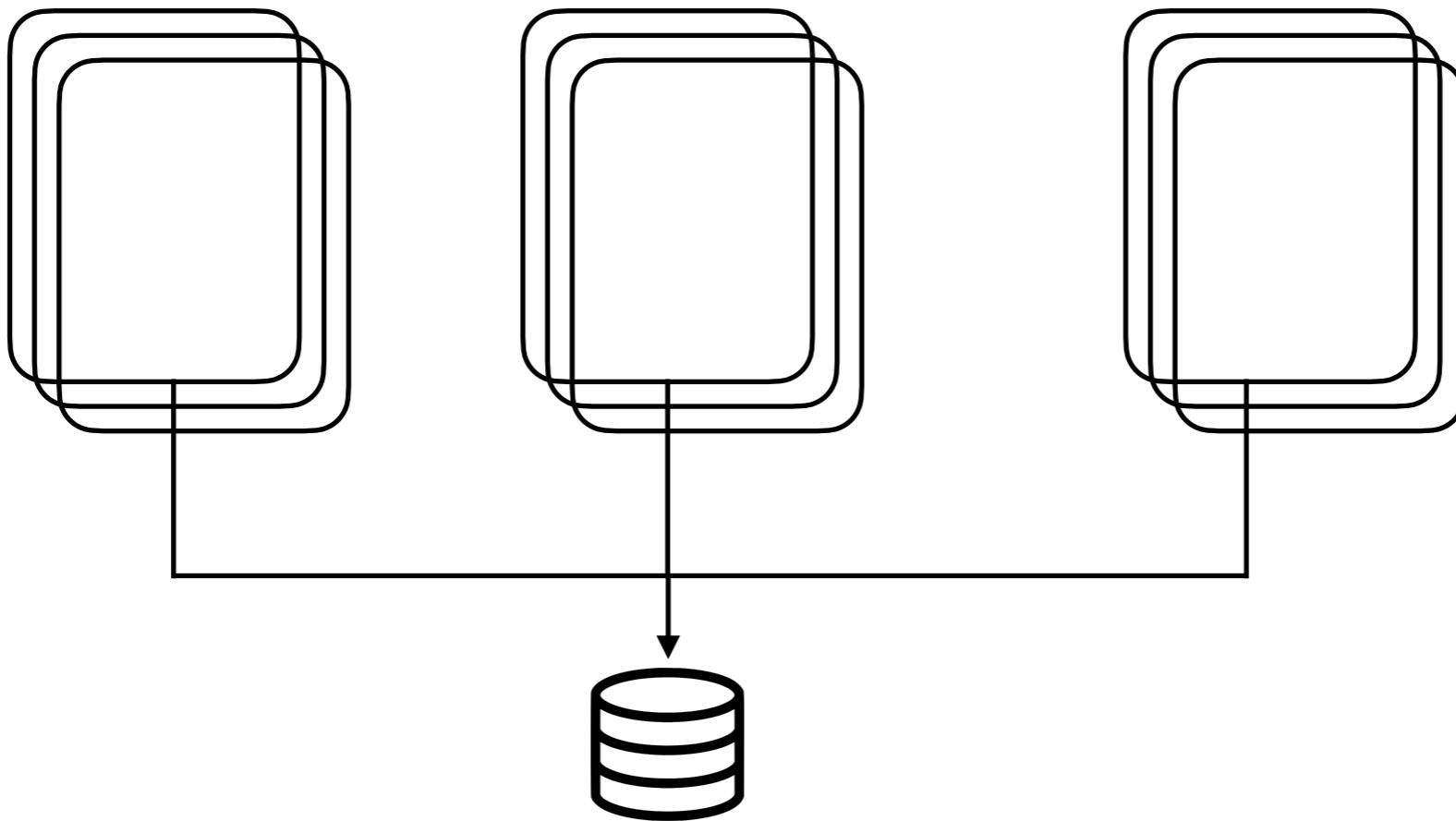




Cube (Split, shard, clone)

API

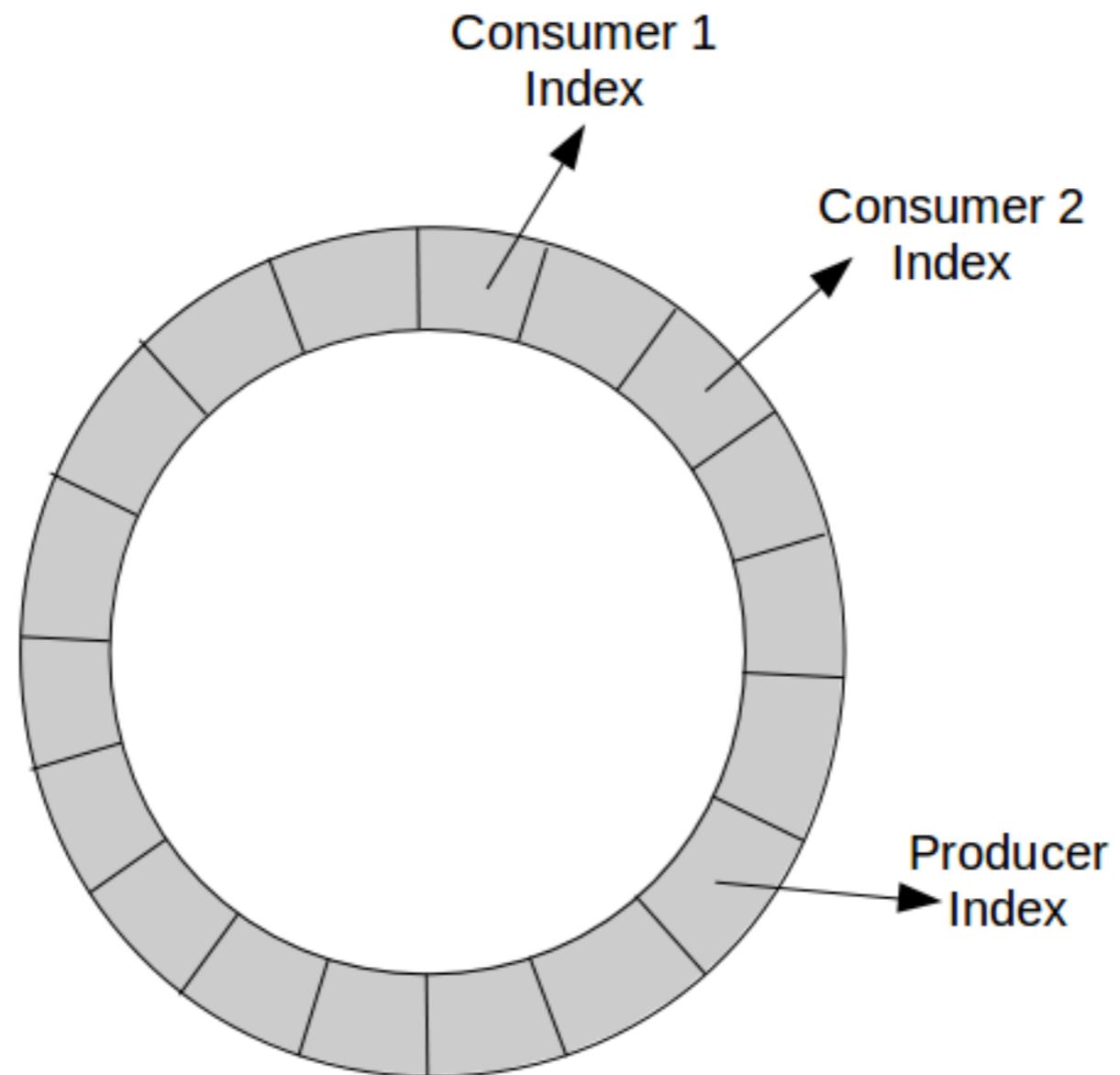




Stock exchange

Case Study

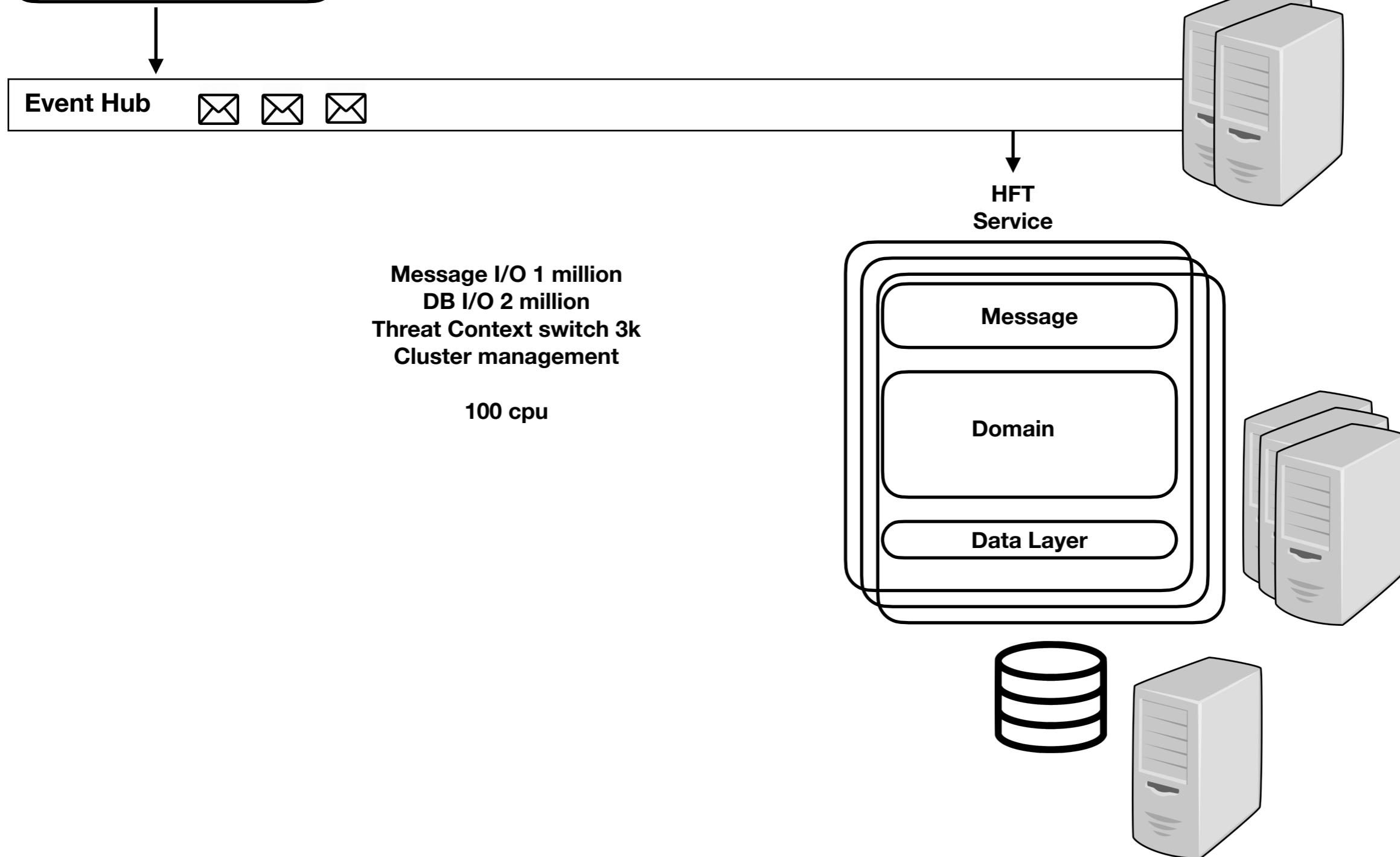
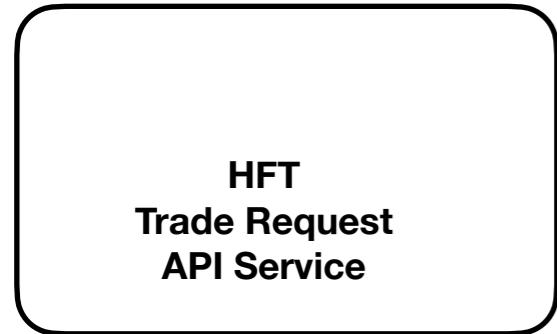
LMAX Disruptor



- I/O are the most expensive operations in a computer
 - Disk
 - Network
 - Database
 - Lock
- Threads are the second most expensive operation on a computer
 - Context switch
-

Cost of I/O Operation

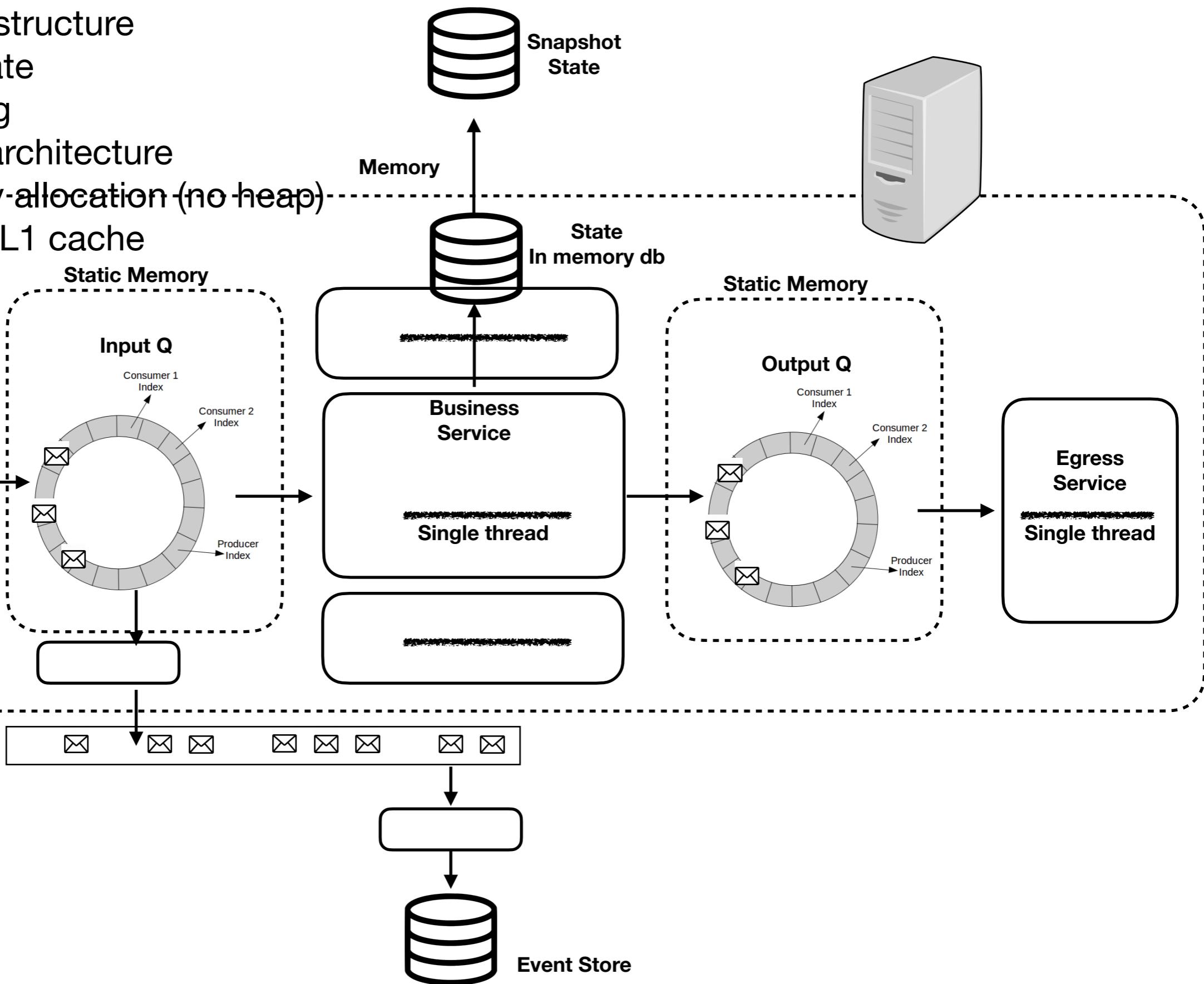
- a+b 3 cpu cycle
- ..
- Create thread 200k cpu cycles
- ...
- Write file 1 million cpu cycles
- Write to db 4 million cpu cycles

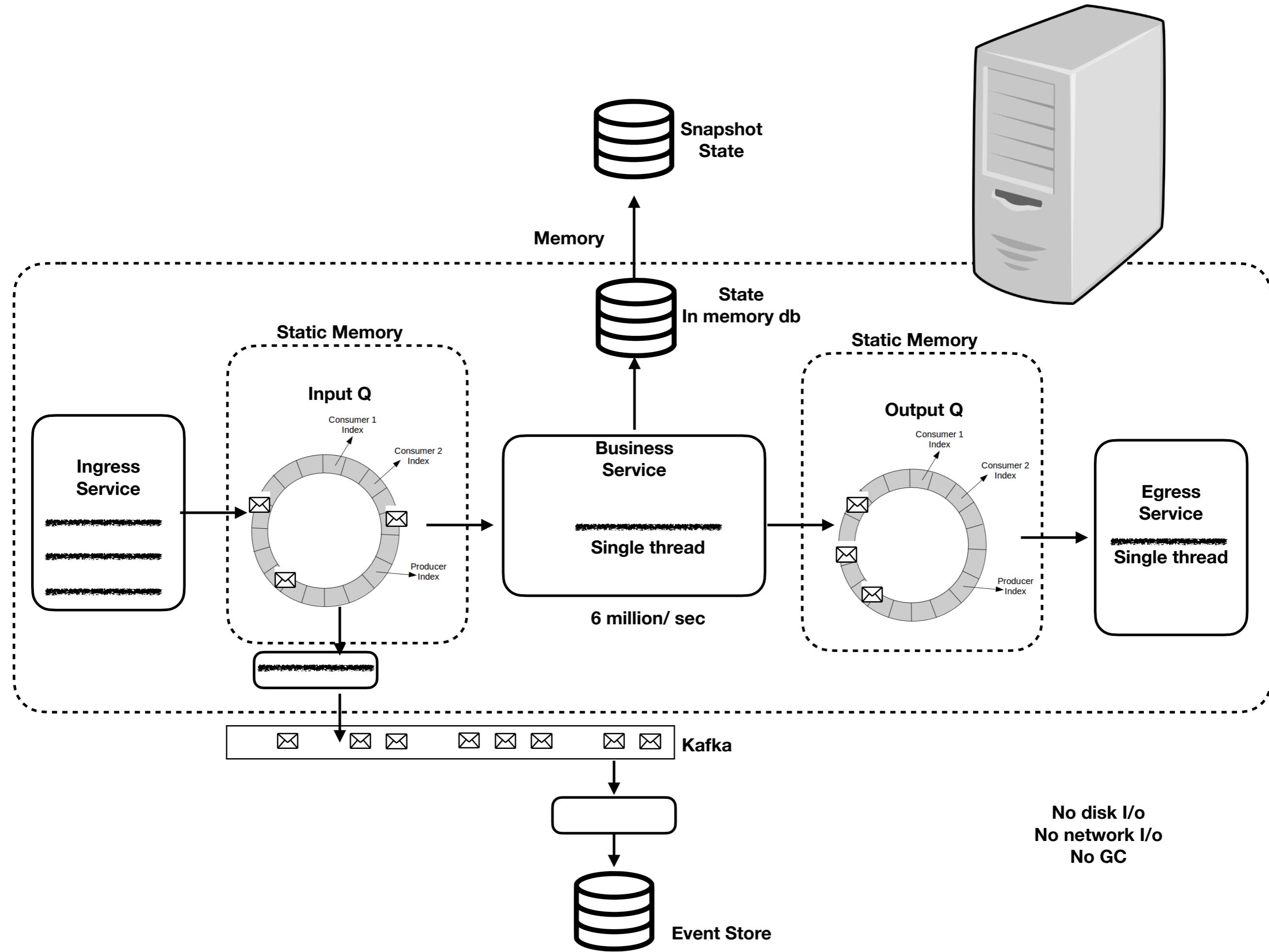


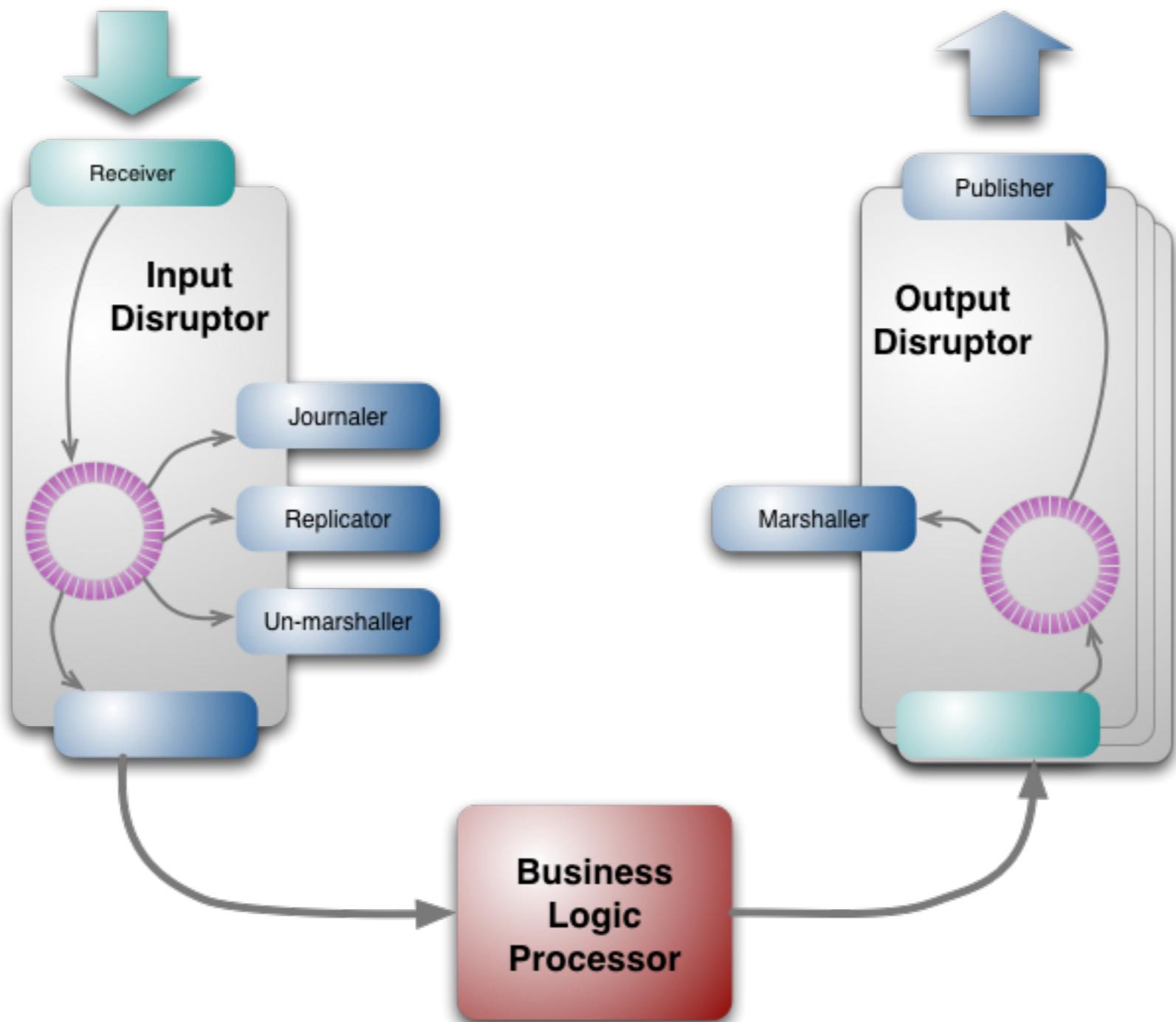
```

# remove I/O operation
# no thread synchronization
# no read writer locks
# efficient data structure
# in memory state
# event sourcing
# Event driven architecture
# static-memory-allocation (no heap)
# leverage Cpu L1 cache

```







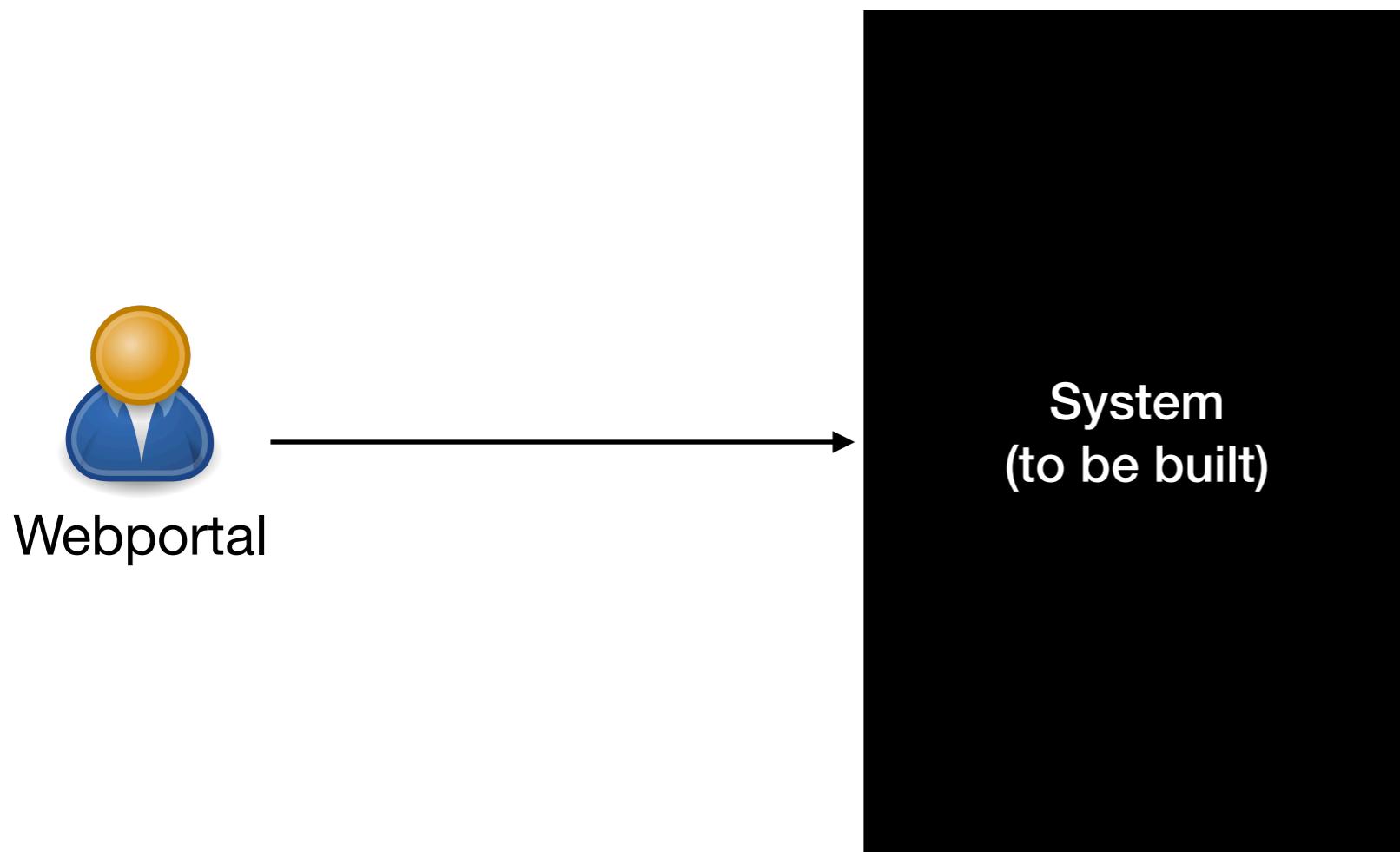
IRCTC Case Study

Architectural Requirements

- # Context View (understand the actors of the system)
- # Functional View (understand major functionality)
- # Constraints (any rules to be followed ?)
- # Quality View (which quality should the system support ?)

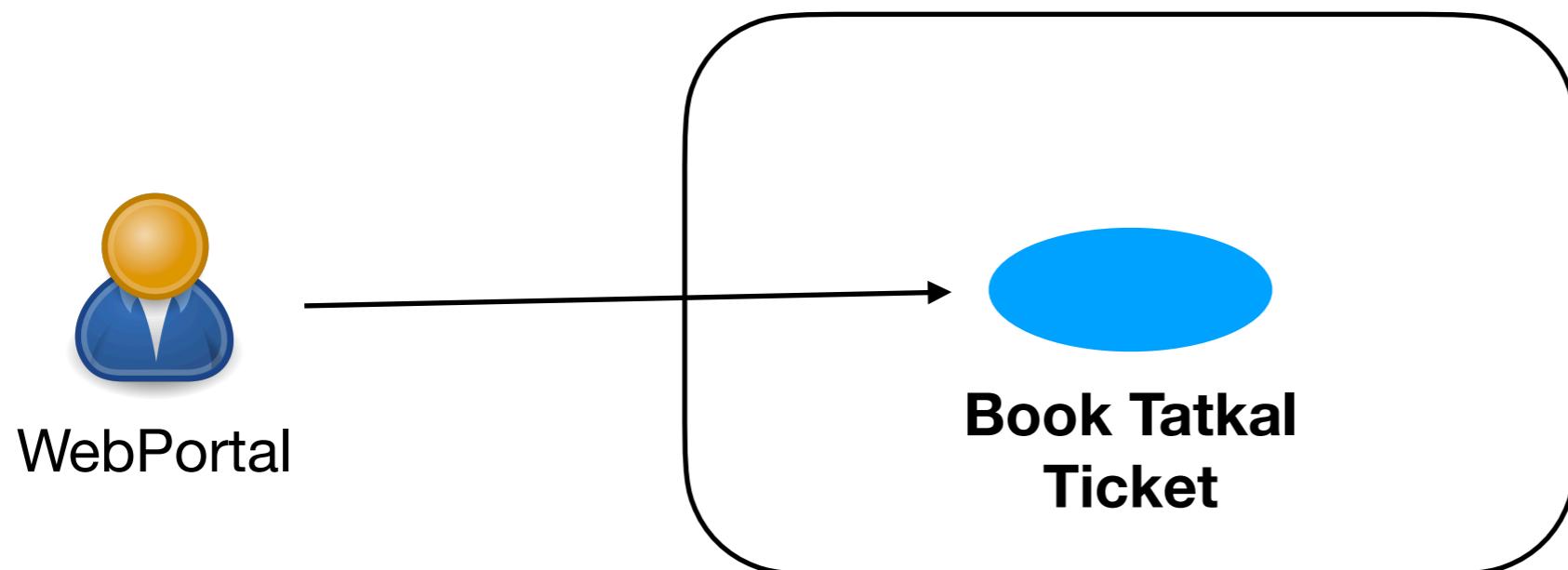
Context View

“Identify all actors”



Functional View

“Identify key functionality”



Constraints

“Identify rules imposed by stakeholders”

- TBD

Quality Requirements

Source (who)	Stimulus (action)	Artifact (which)	Environment (context)	Response (output)	Measure (scale)
A user	booking a tatkal ticket	On portal/ App	during Tatkal time (10 AM to 12AM)	Should book ticket	In 2 seconds.

Architectural Design

- # Logical View (decomposition decisions)
- # Deployment View (deployment decisions)
- # Security View (Security decisions)
- # ...

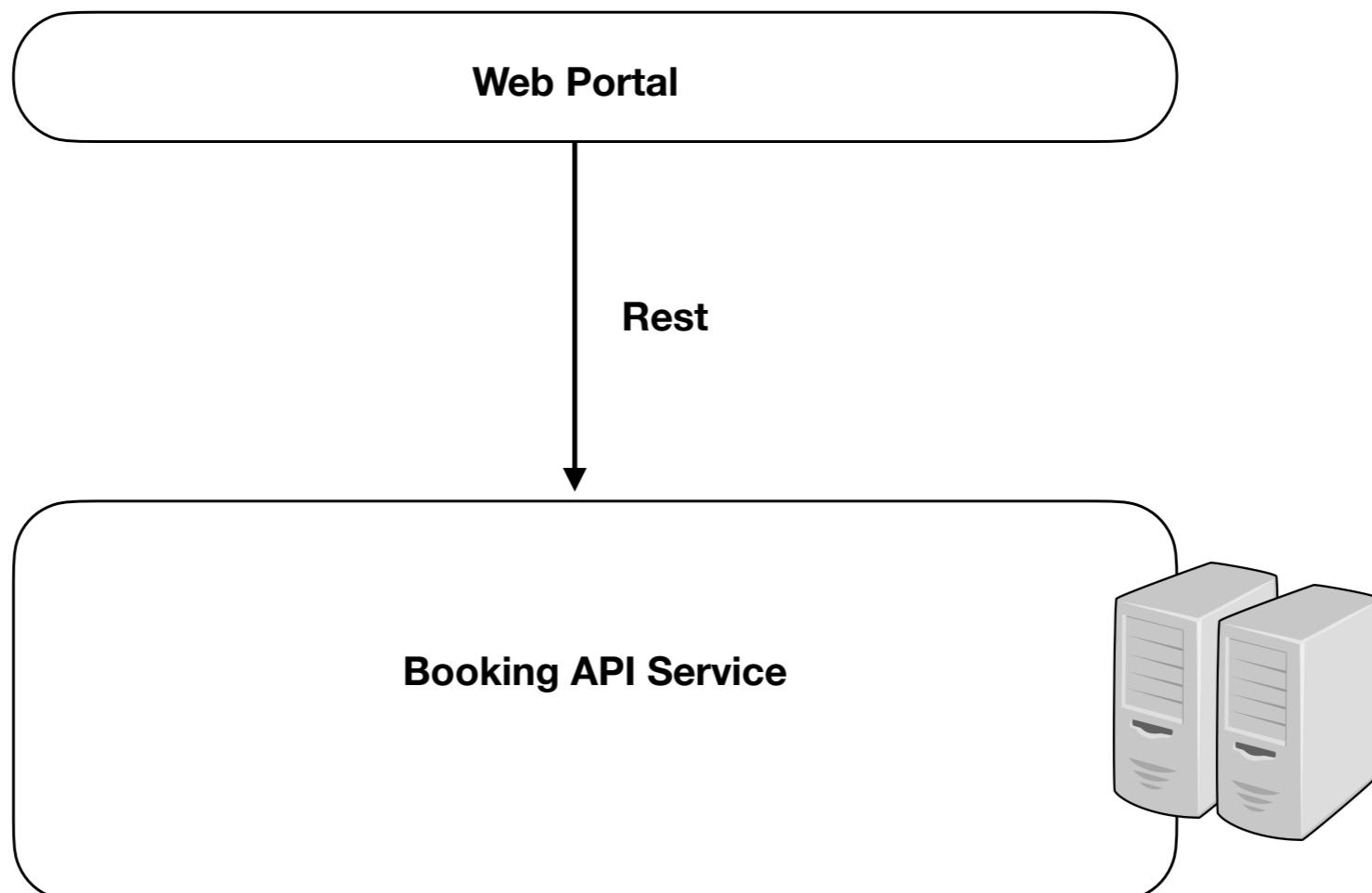
Approach

- Limited features during tatkal
- Time slicing functionality
- Sharding
 - Region
 - Types of coach,
- Cache
 - Train information
- Virtual queue
- Block automated booking
- History data - wide column (cassandra)
-
-
-
- Eventual consistency
- Optimistic locking
-

- Immediate (redbus, irctc)
 - Pessimistic locking
 - Optimistic locking
- Eventual (amazon, agoda, payments)
-

Logical View

- # System Decomposition**
- # Persistence approach**
- # Compute approach**
- # Communication approach**
- # cross cutting approach**



Current
booking

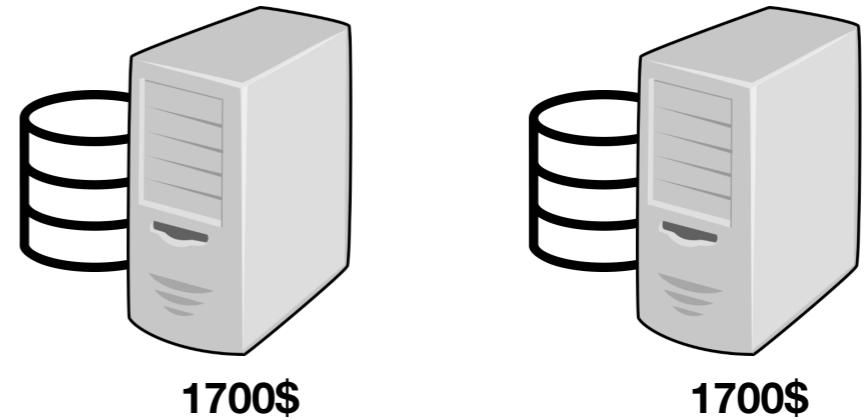
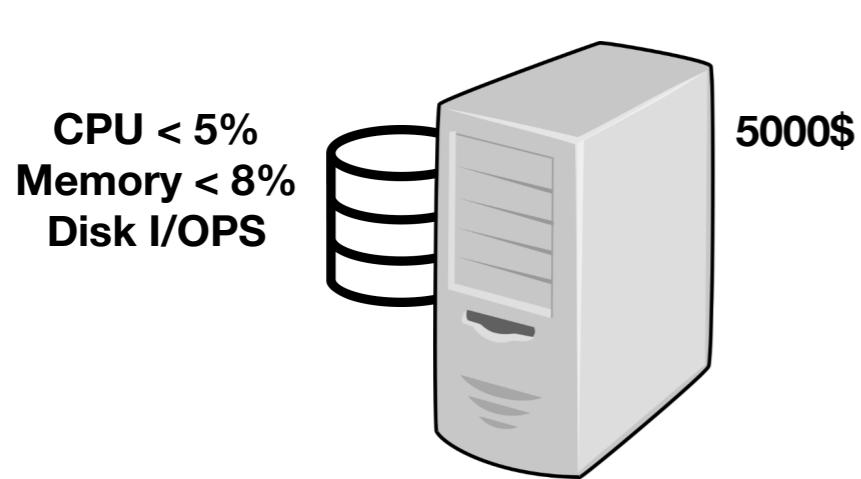
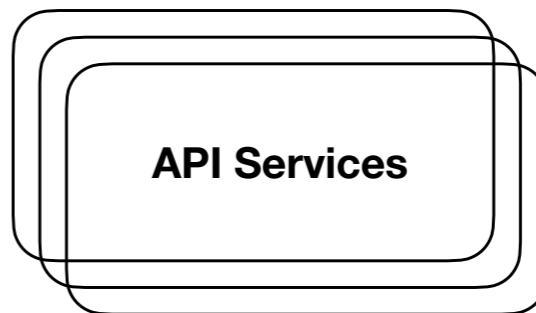
Rdbms



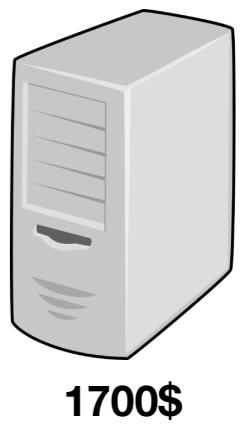
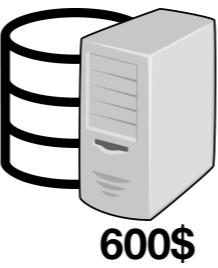
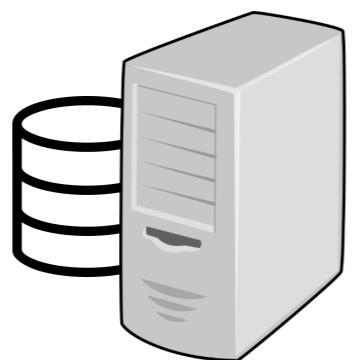
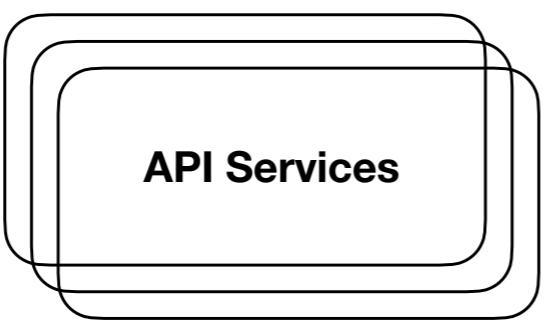
Completed
booking

Wide column

IOPS
numbers of rows (size of storage)
query complexity
Concurrency
Capacity planning

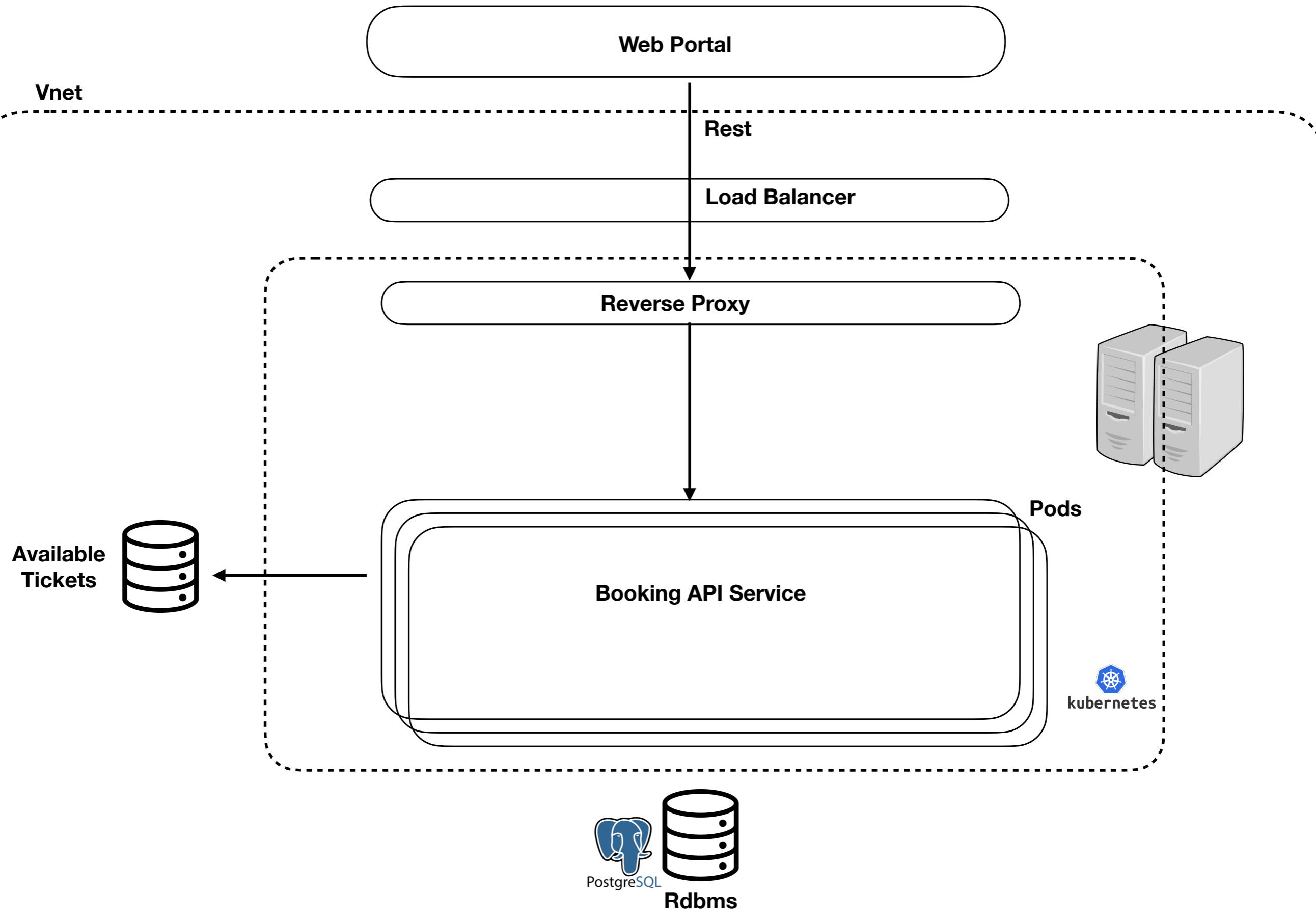


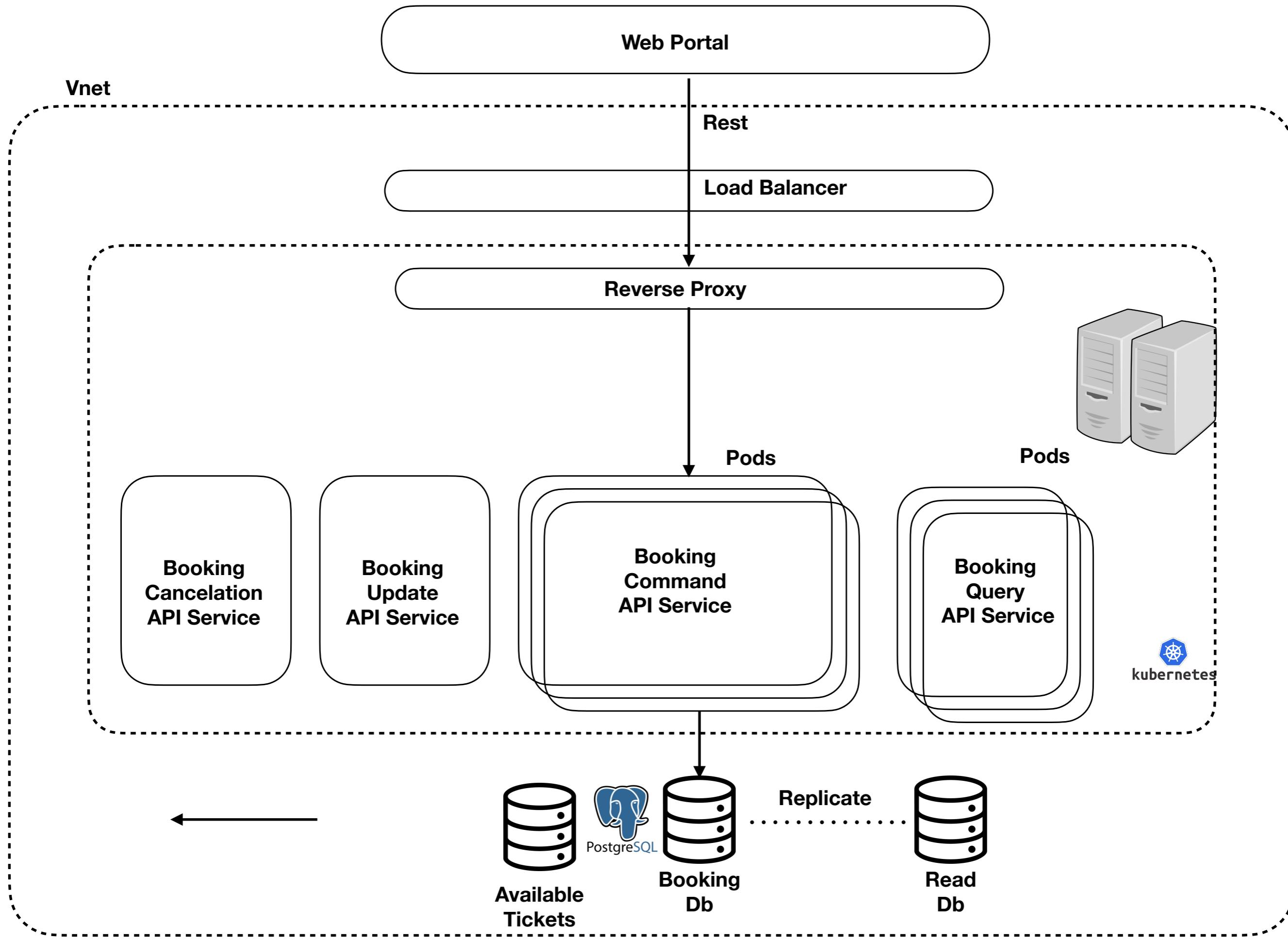
2 vcpu, 4gb \$ 70
4, 8 \$
...
94vcpu, 340, \$6500



How to scale booking db

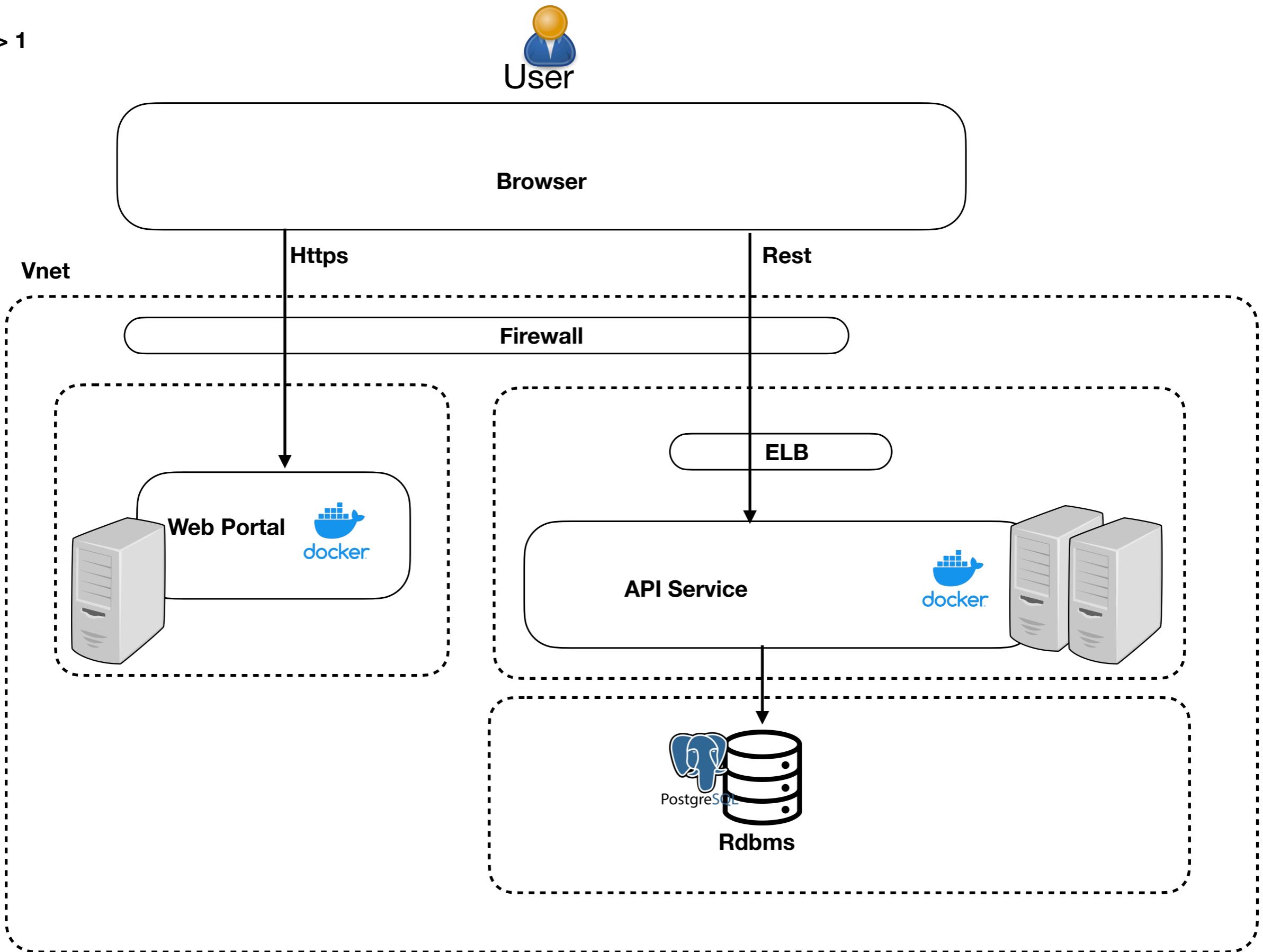
- Shard by ticket type (Tatkal , general)
-





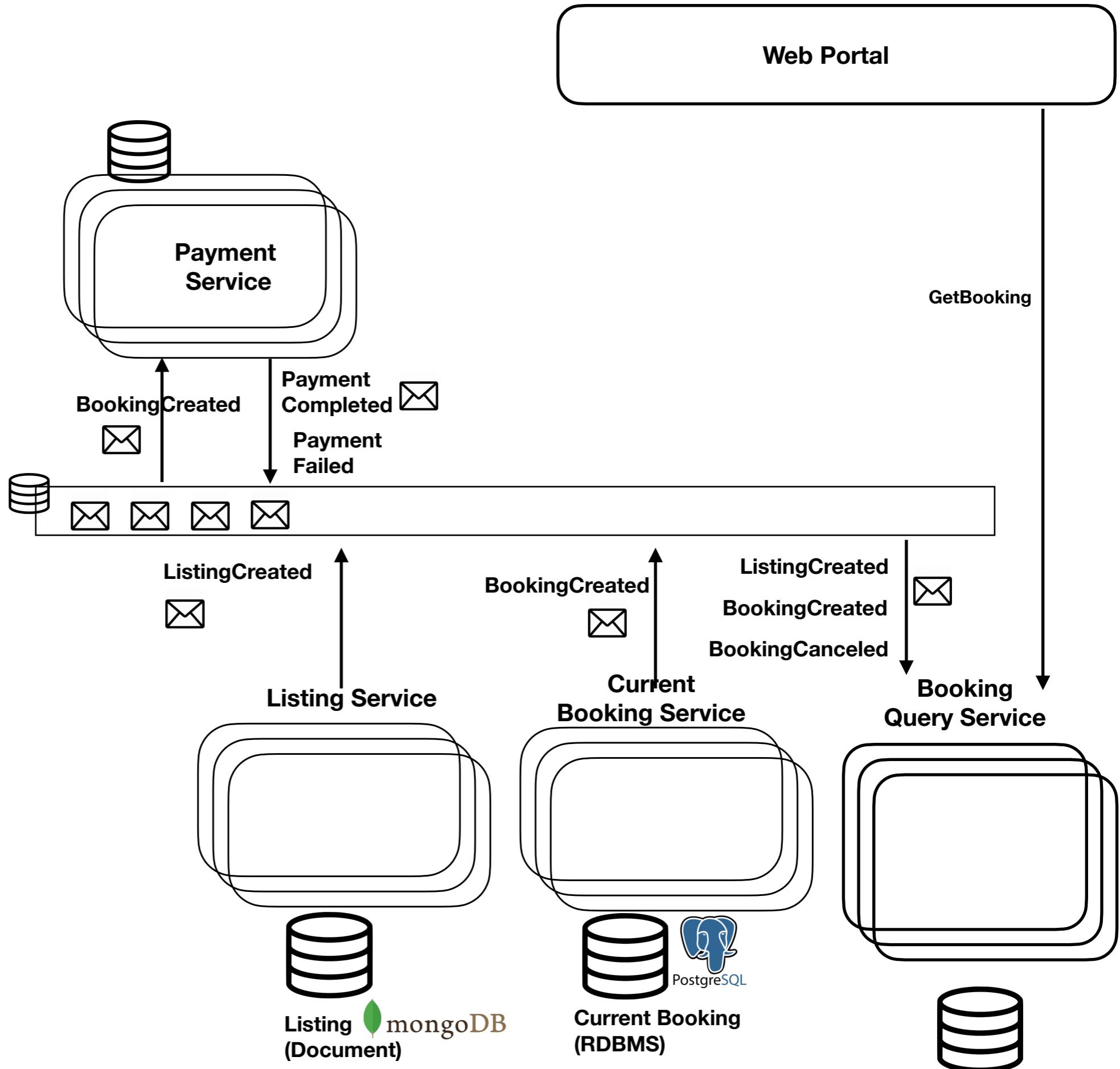
Deployment View

4 -> 1



Architectural Eval

```
# Domain Command  
# Domain Events  
# Domain Exception
```



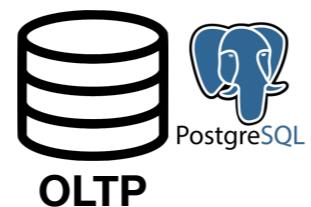
windsurf



Trae

Zed

Unit Test/ Integration Test
- generative tools



Web Portal



Reports



Command API

5 %

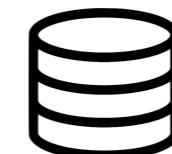


Query API

95 %



Power BI Server



OLAP

ClickHouse

