Software Engineering Conference Russia 2018

October 12-13
Moscow

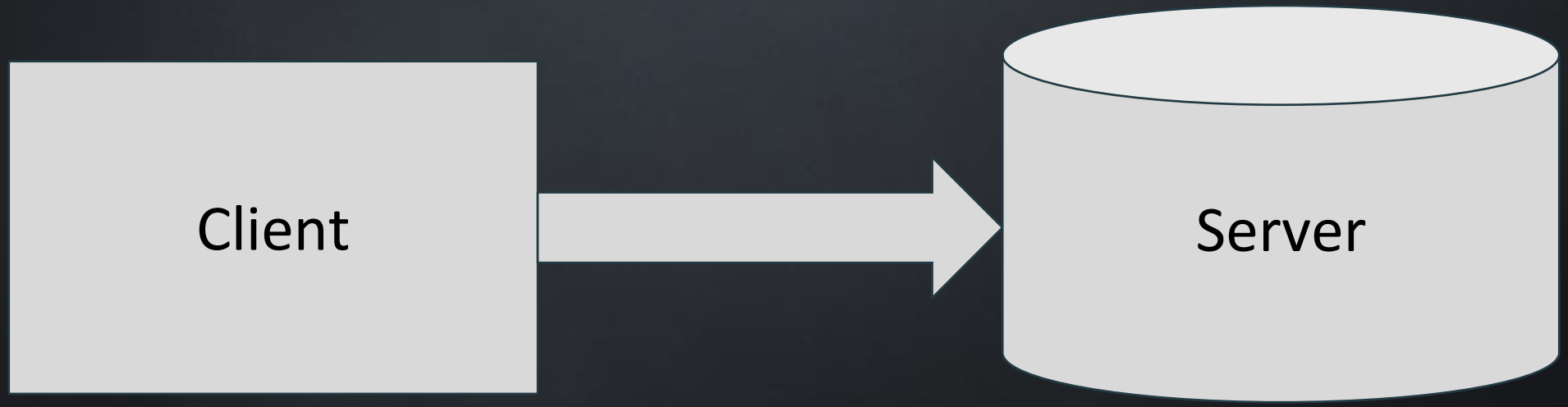
Why microservices do not fly and how to help them to take off

Alexandr Shcherbakov

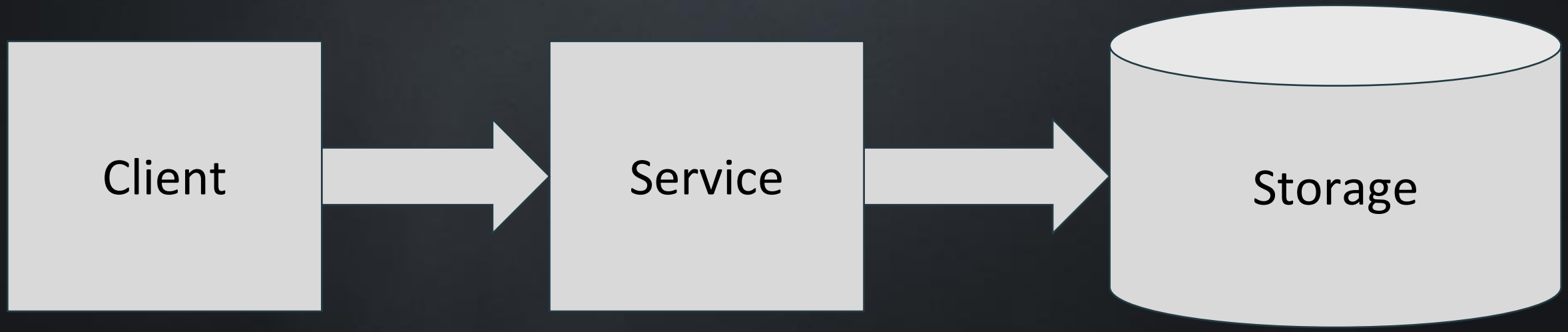
Auriga, Inc

Quiz

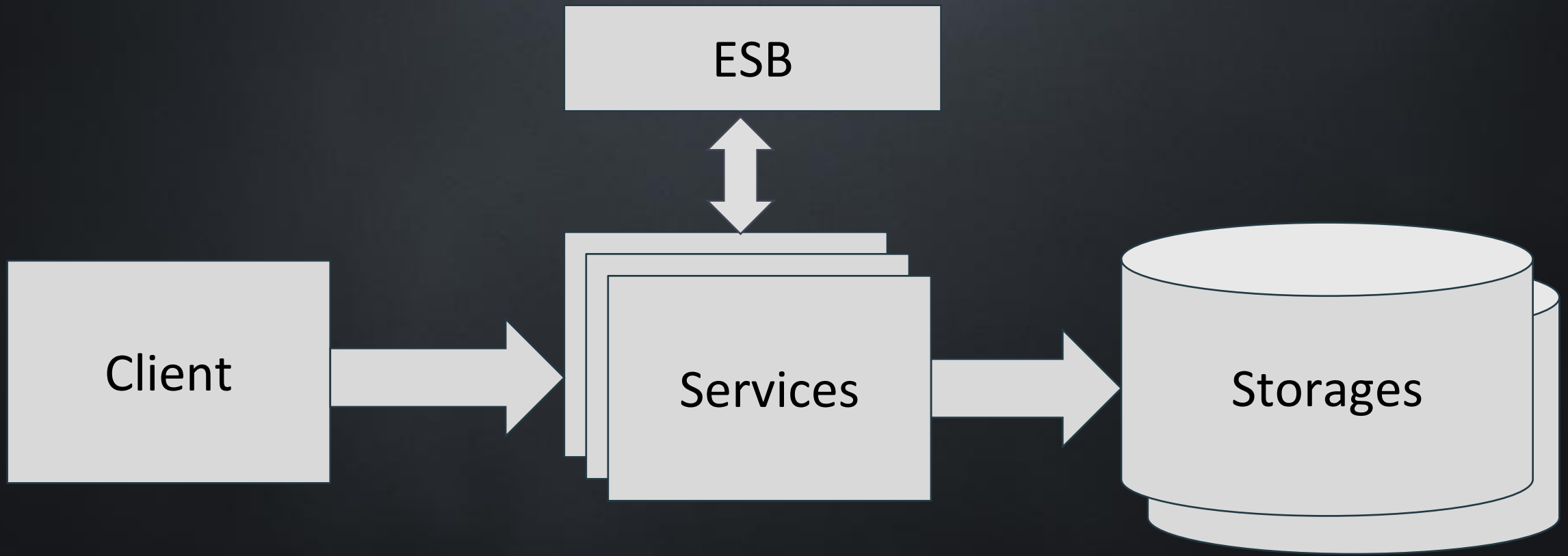
- Who has ever developed microservices architecture?
- Who has positive experience?
- Who has negative experience?



Classic client-server architecture



Multi-tier architecture



SOA

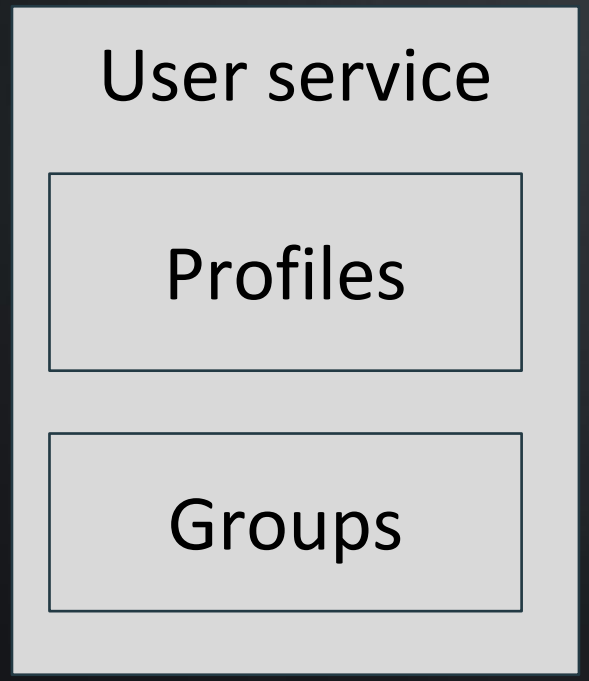
Nowadays typical case

1. We need a new enterprise system
2. We have the distribution network of clients
3. We have only 6 months for MVP and one year for first release, developing for the end of days.
4. But microservices are strongly required!

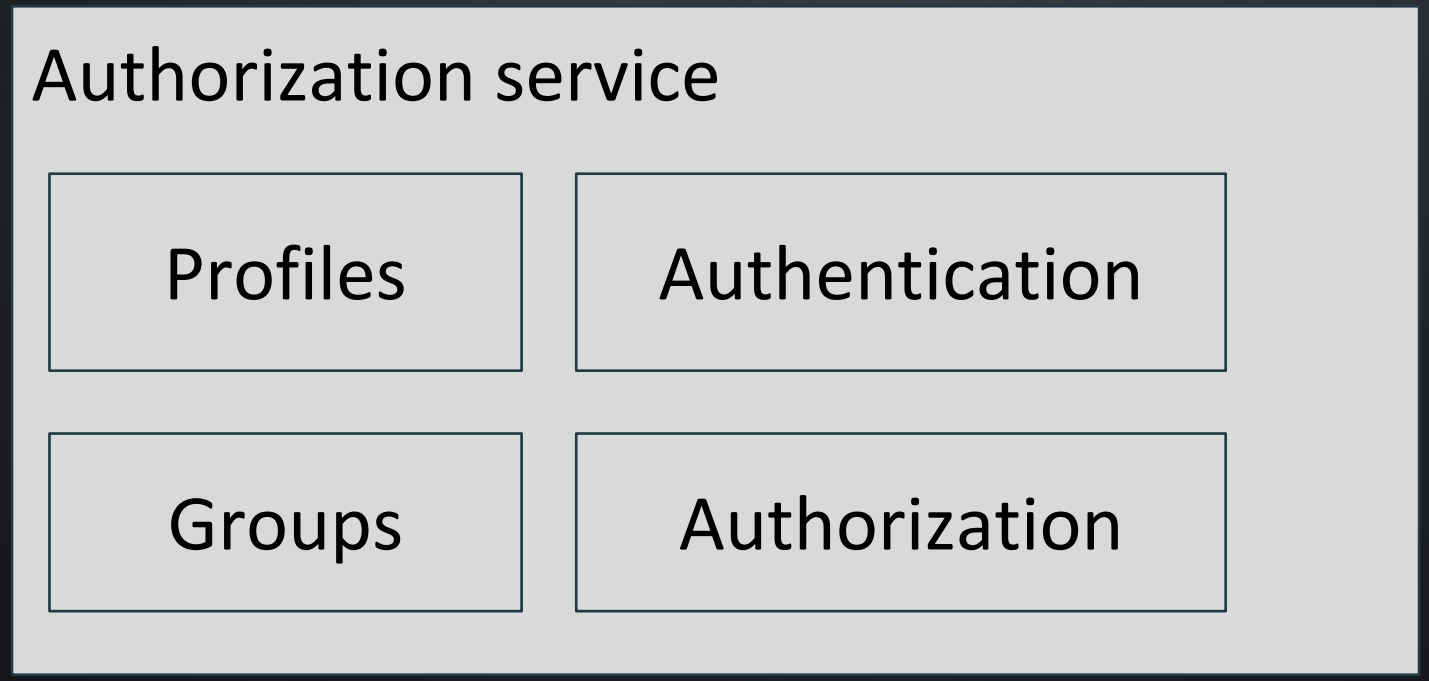
Microservices are just services but...

- Domain splitting
- Independence
- Everyone has own storage

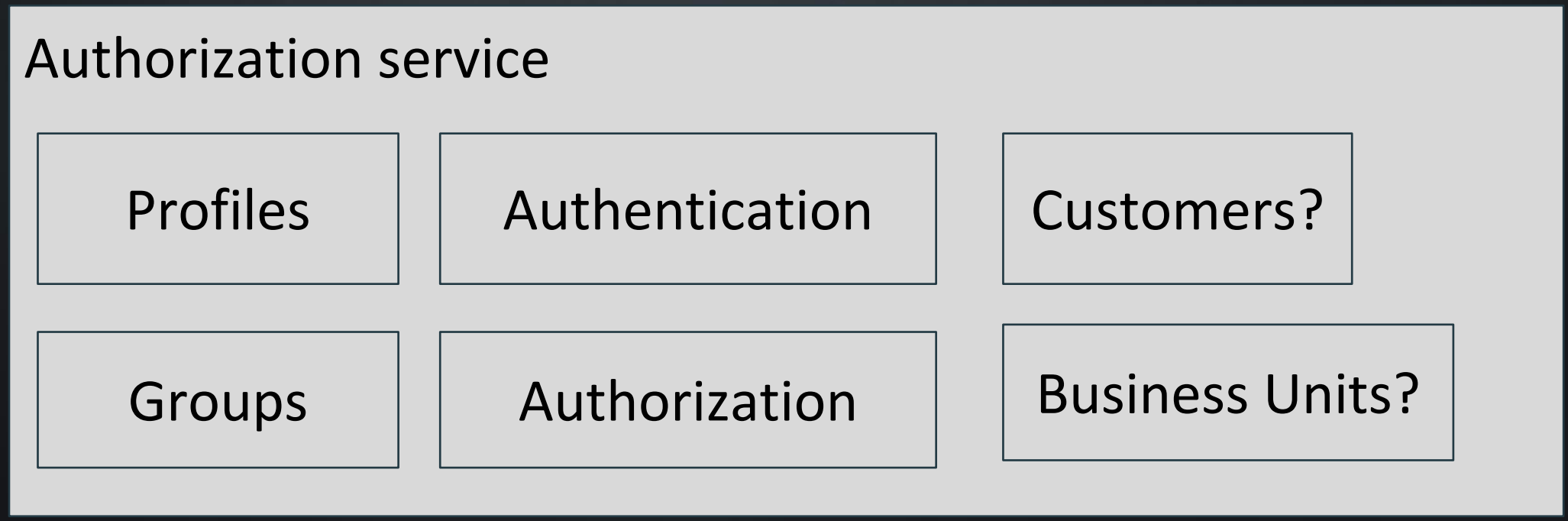
How to name a microservice?



How to name a microservice?



How to name a microservice?

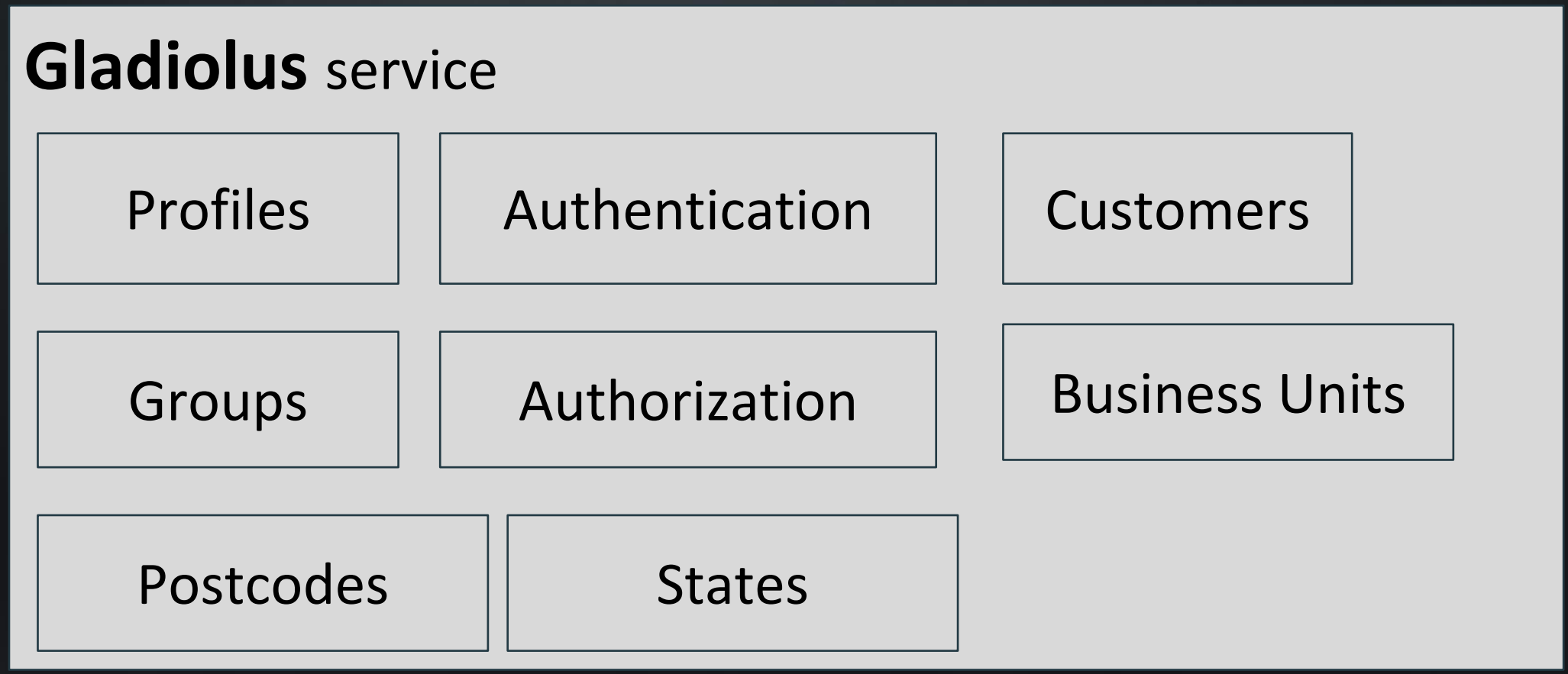


How to name a microservice?

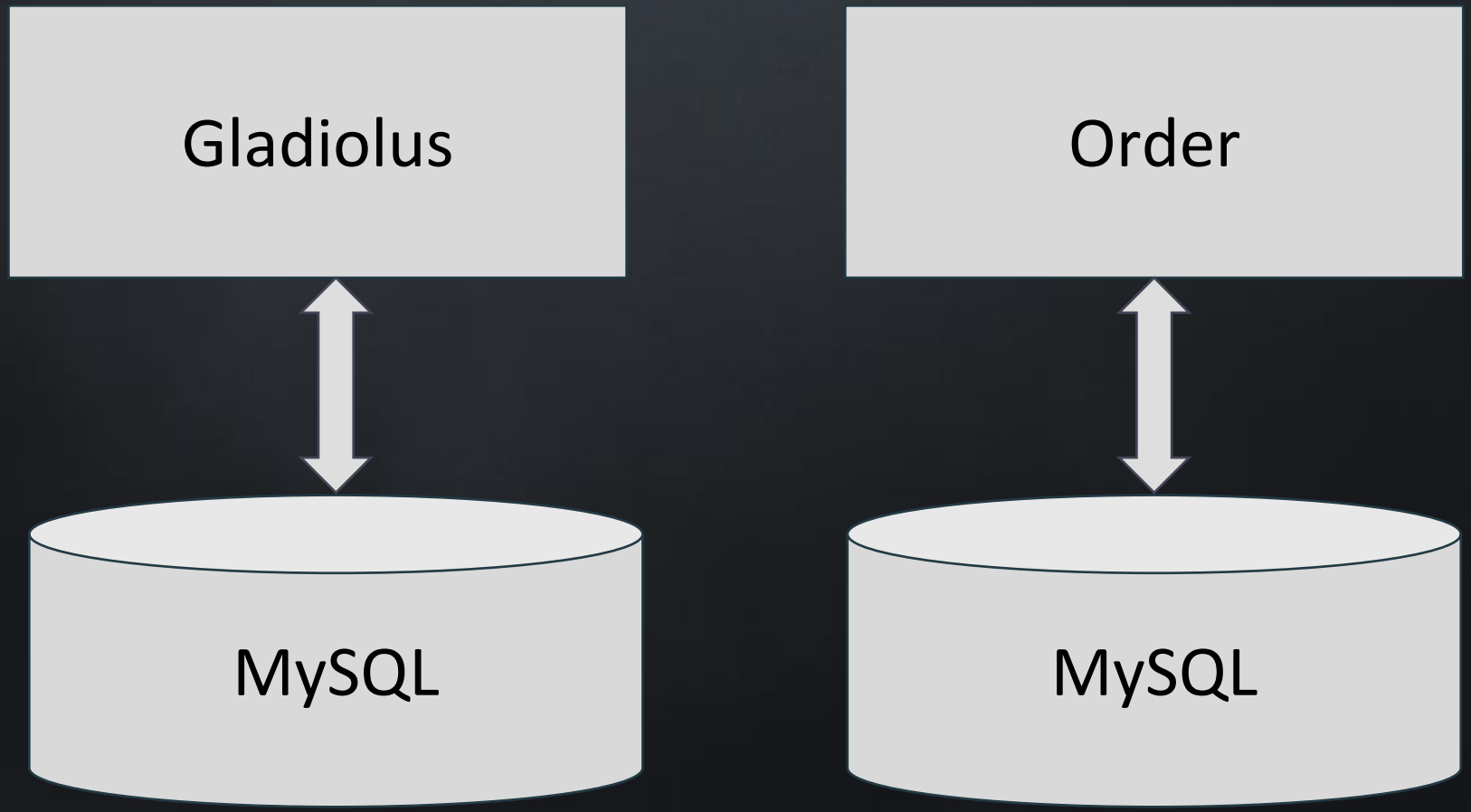
Gladiolus service

Profiles	Authentication	Customers
Groups	Authorization	Business Units

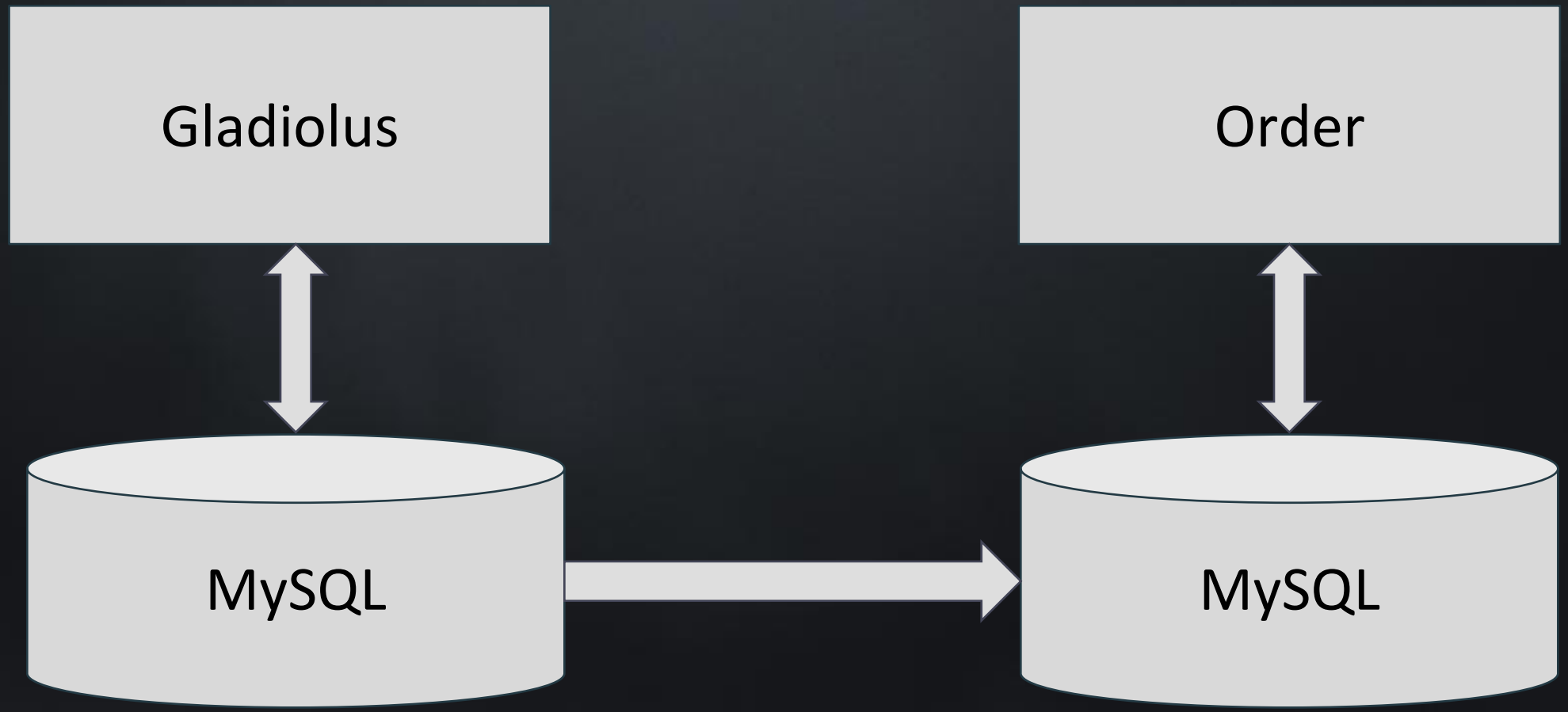
How to name a microservice?



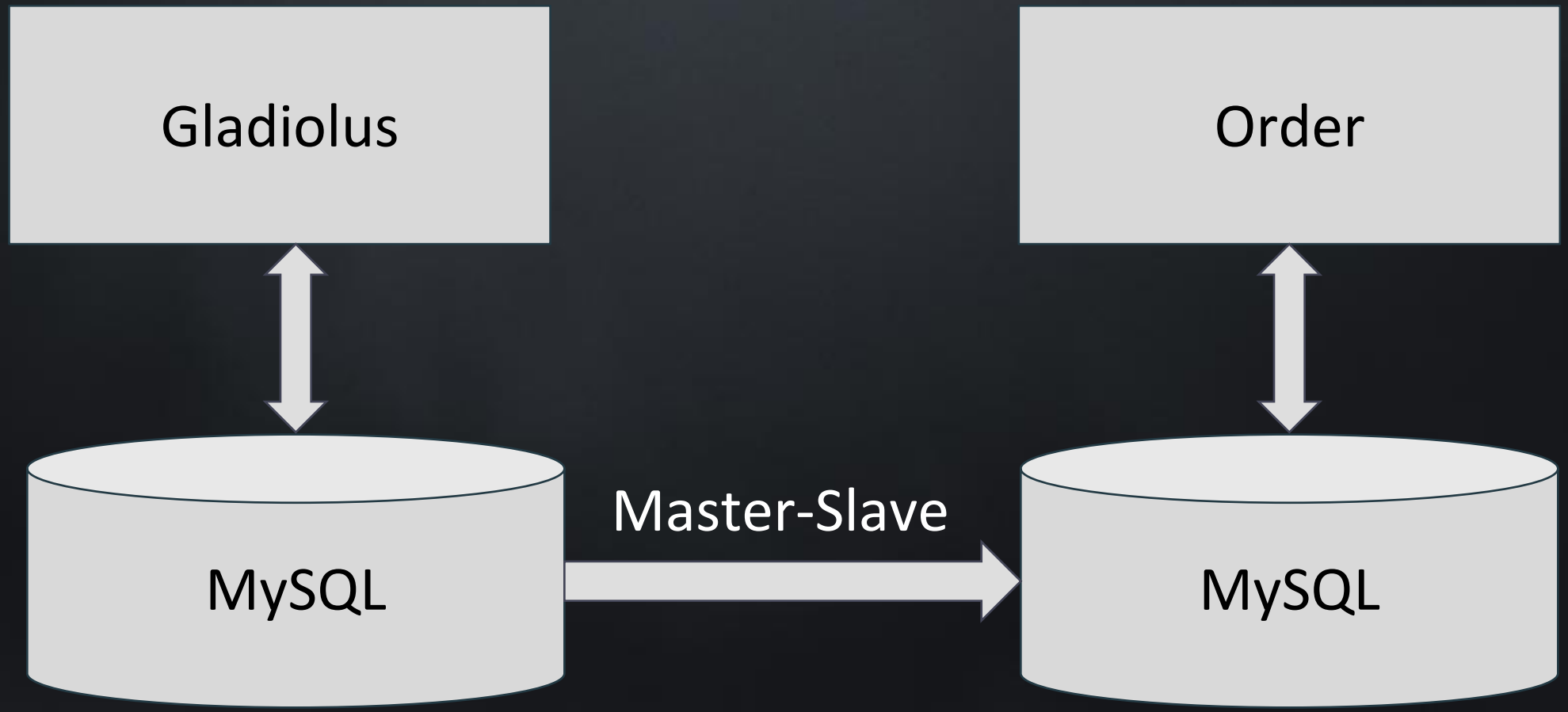
How to synchronize data storages?



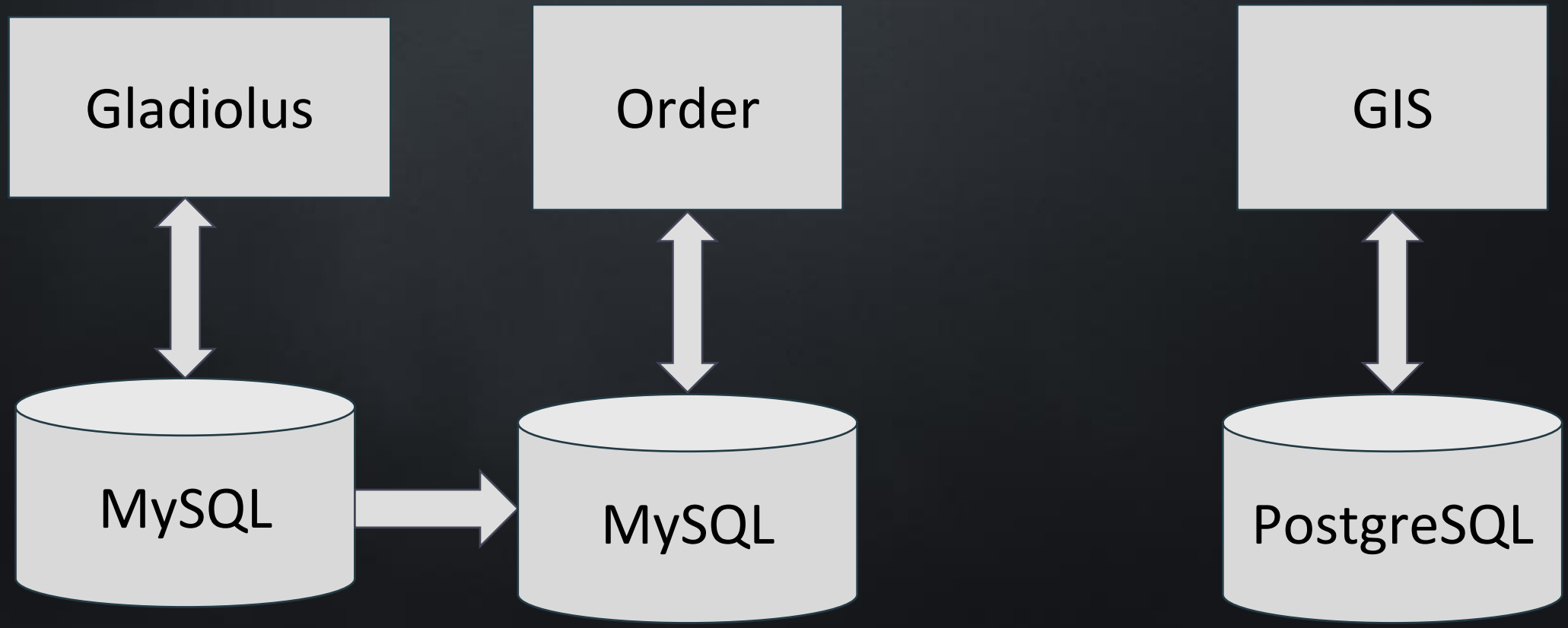
How to synchronize data storages?



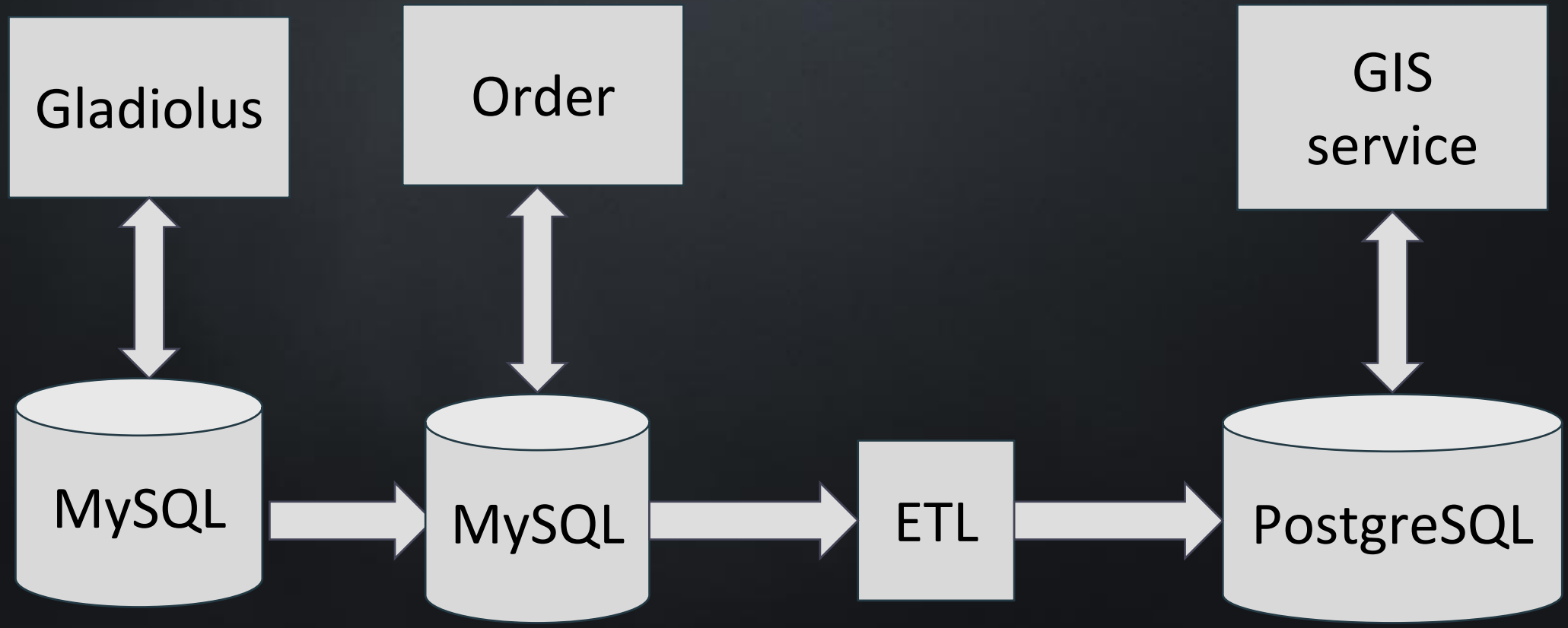
How to synchronize data storages?



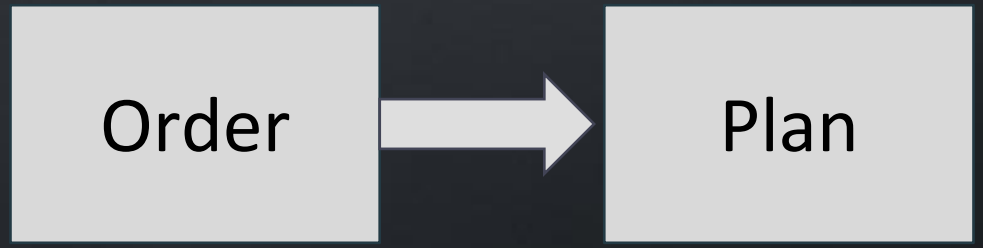
How to synchronize data storages?



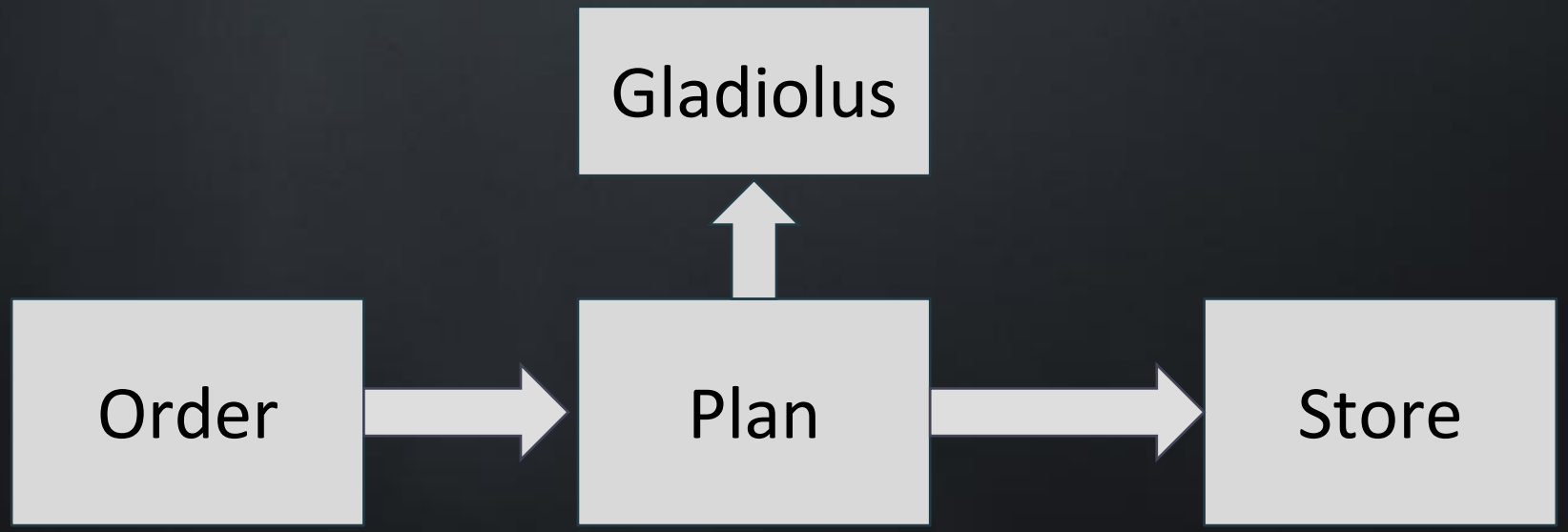
How to synchronize data storages?



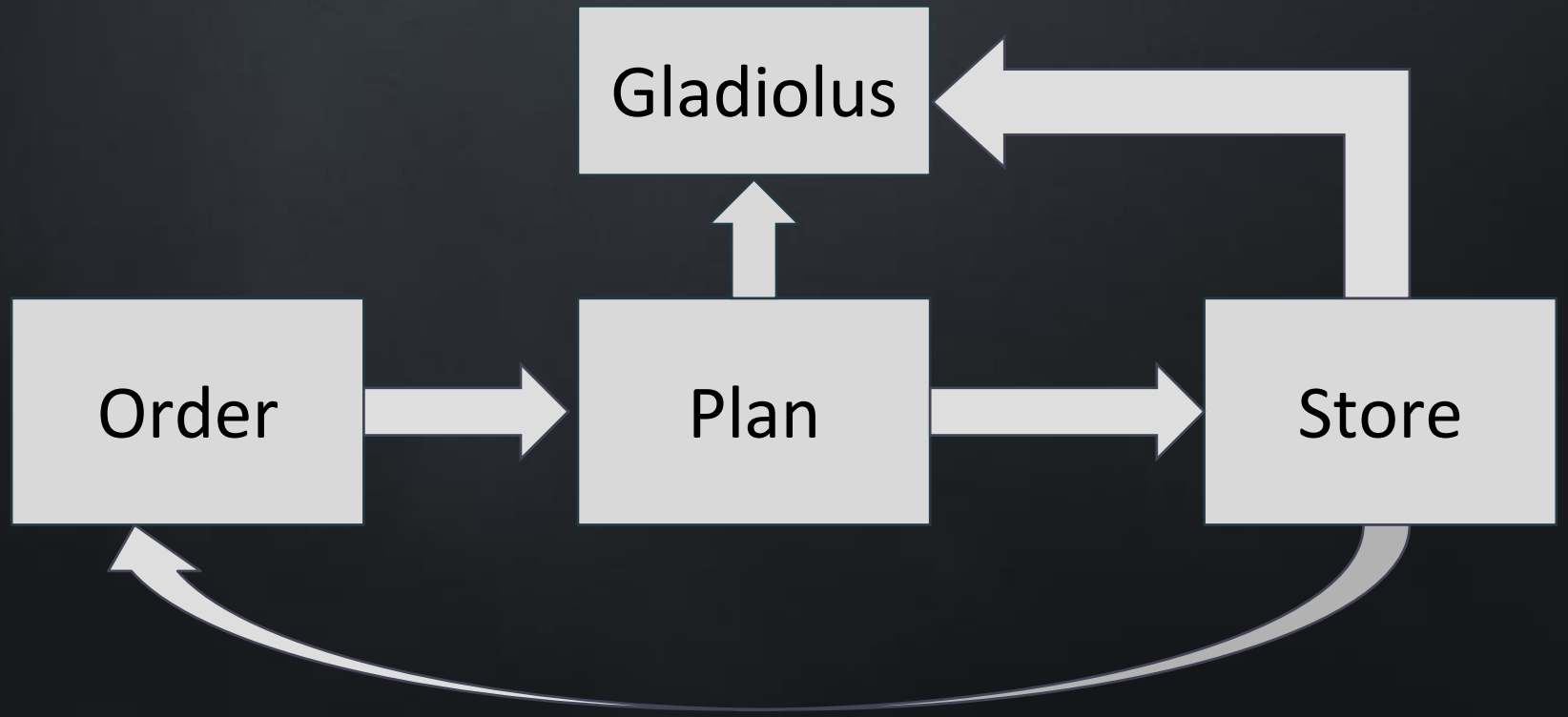
How to call a service?



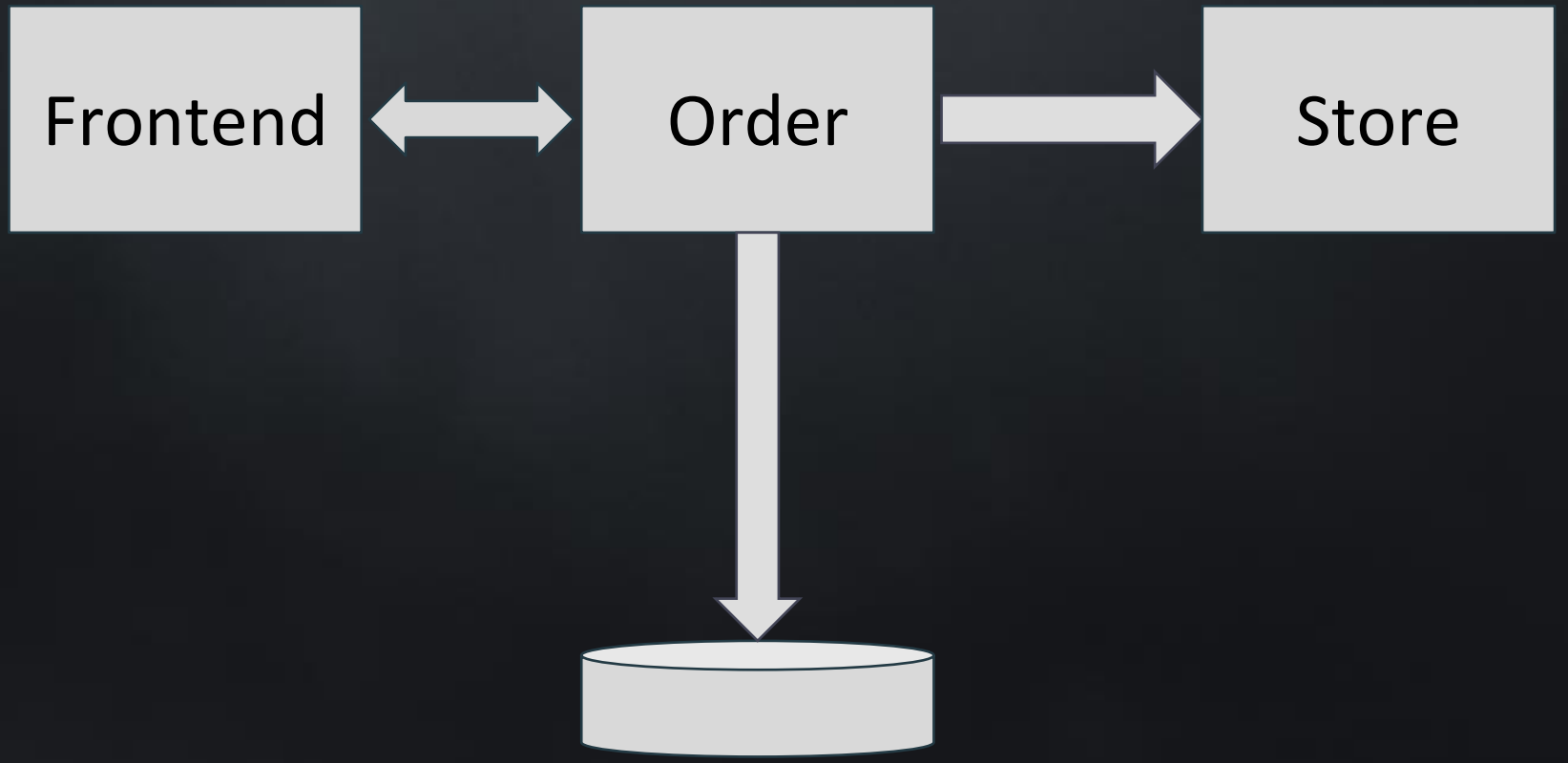
How to call a service?



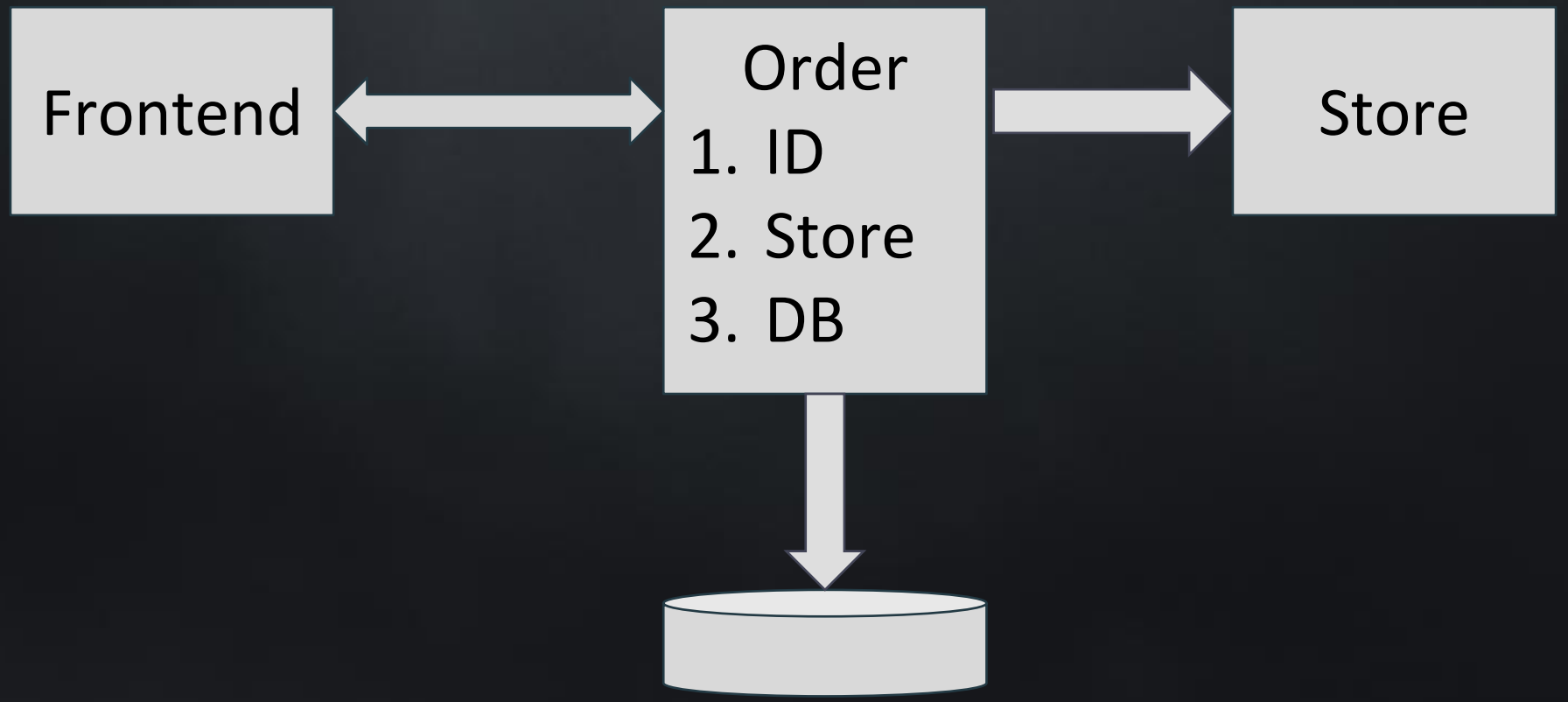
How to call a service?



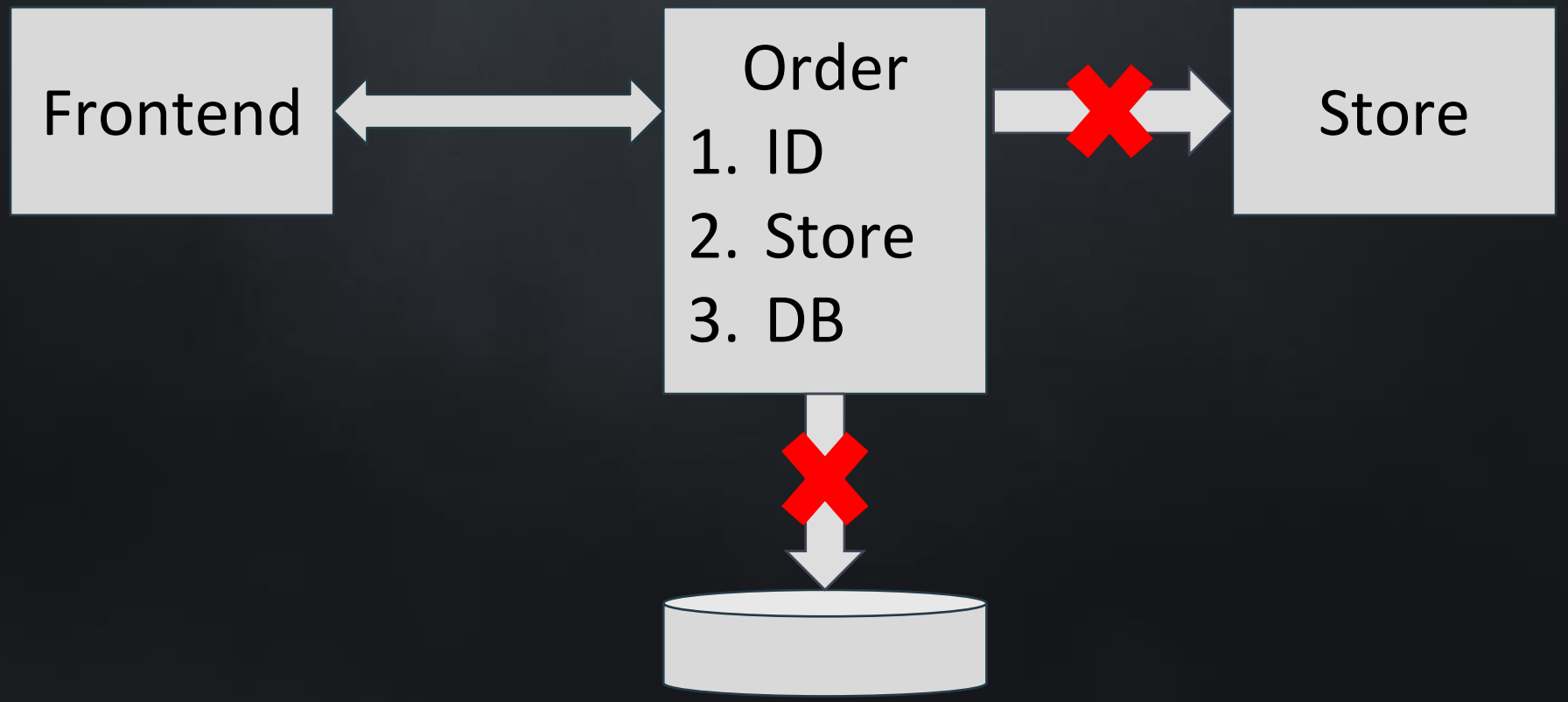
How to do transactions?



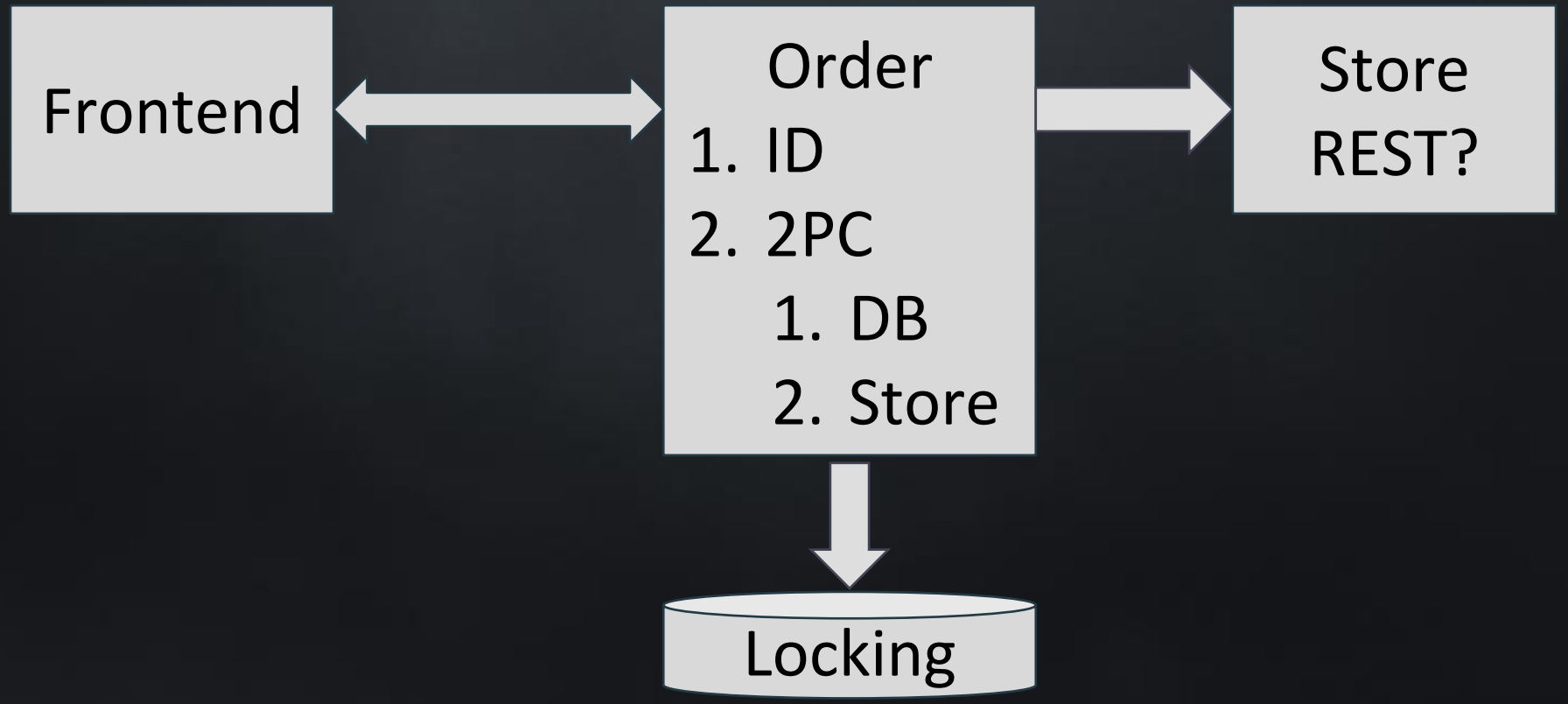
How to do transactions?



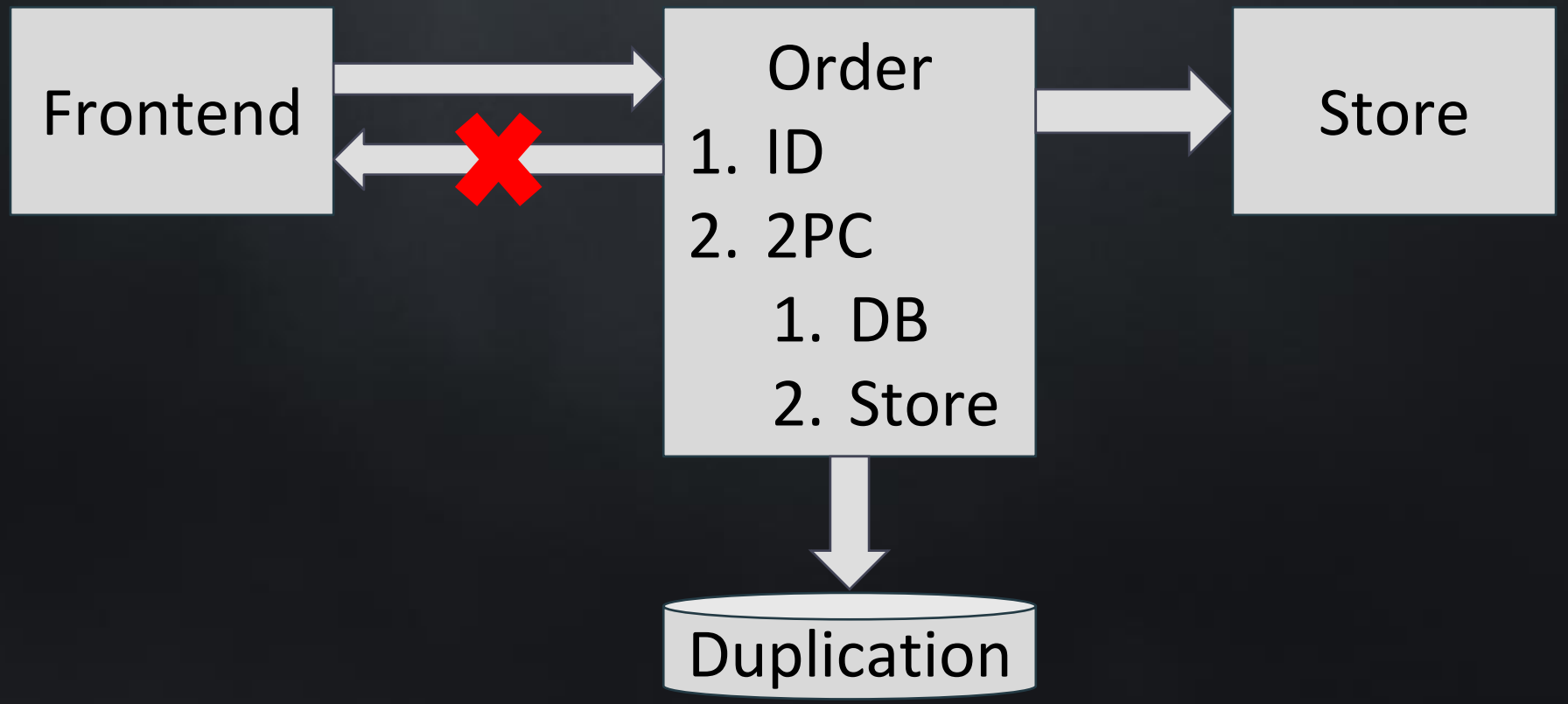
How to do transactions?



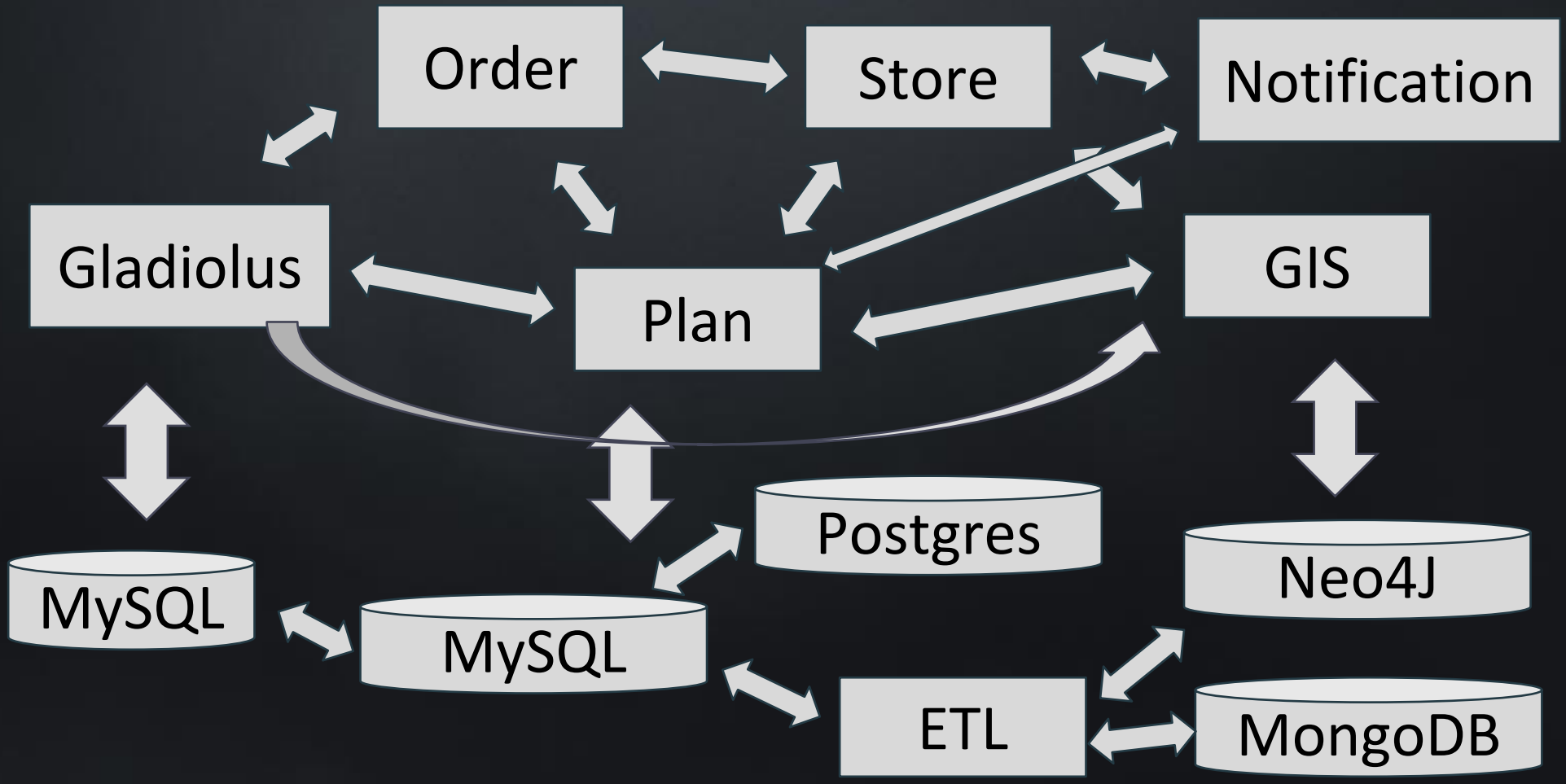
How to do transactions?



How to do transactions?



How to design software architecture?



How to change software architecture?

Top 3 of microservices benefits

- Flexible scaling
- Flexible deployment and functionality
- Flexible development

The key benefit of microservices architecture

Flexibility!

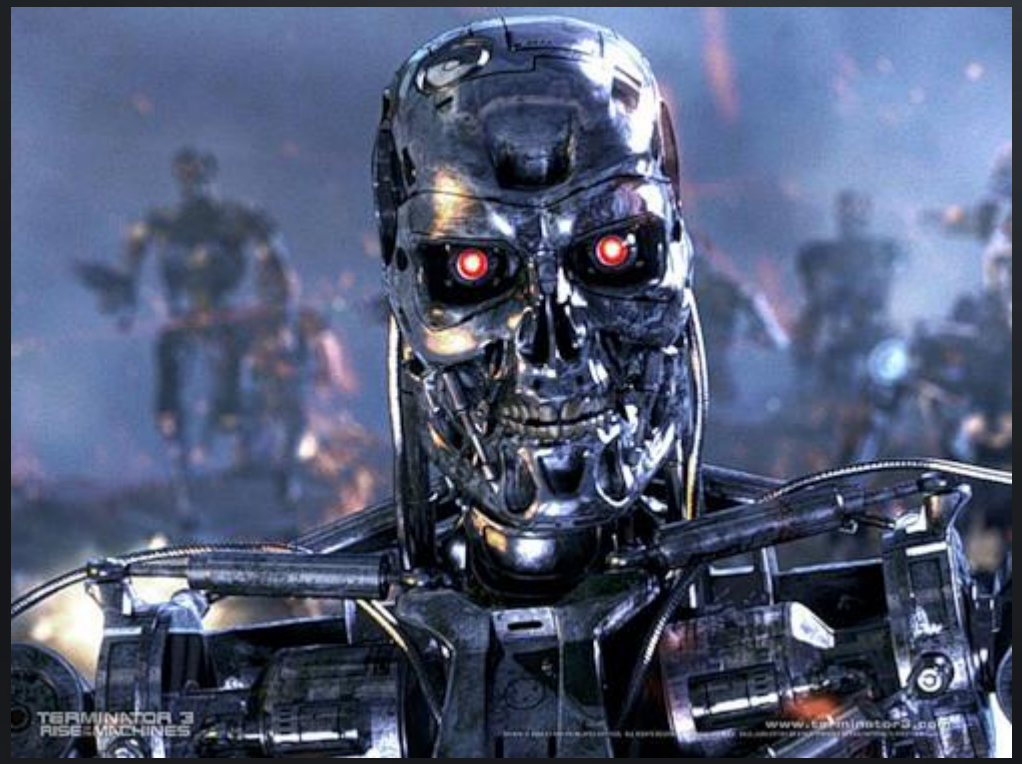
Agile!

Monolith vs Microservices

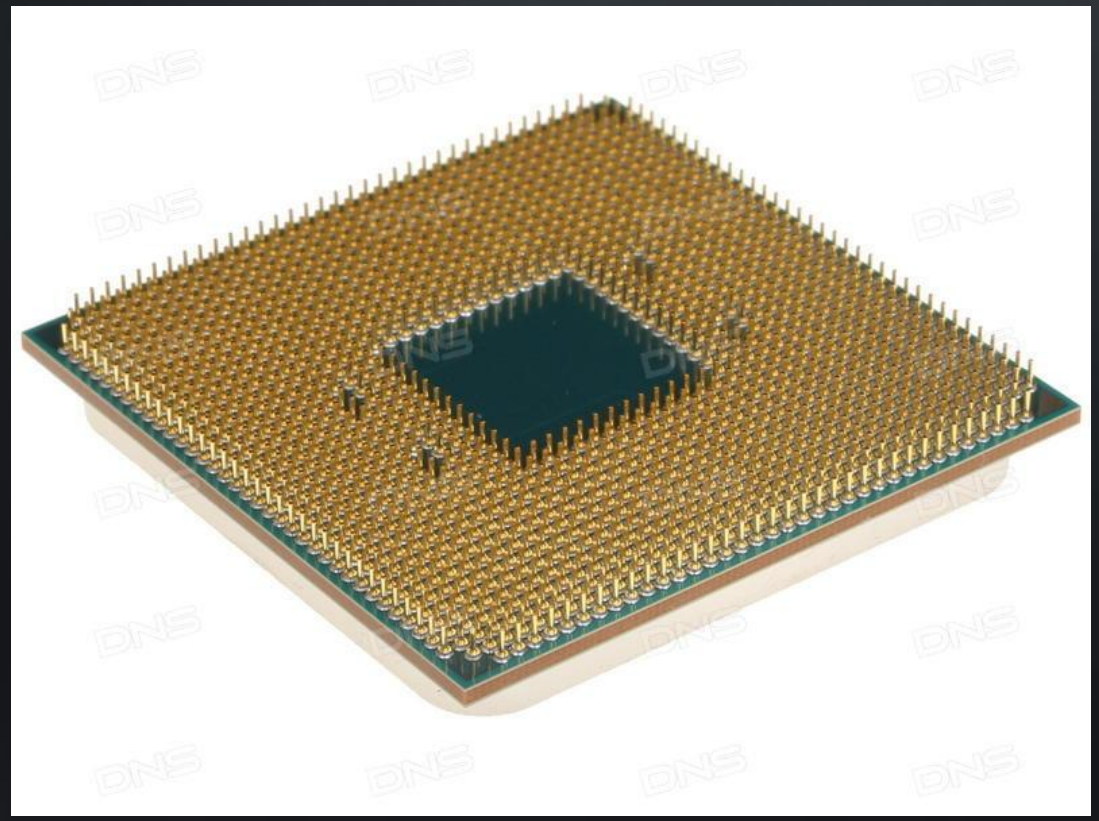
- Clear and hard logic. ACID
- Strong contracts and interfaces
- Integrity and normalization

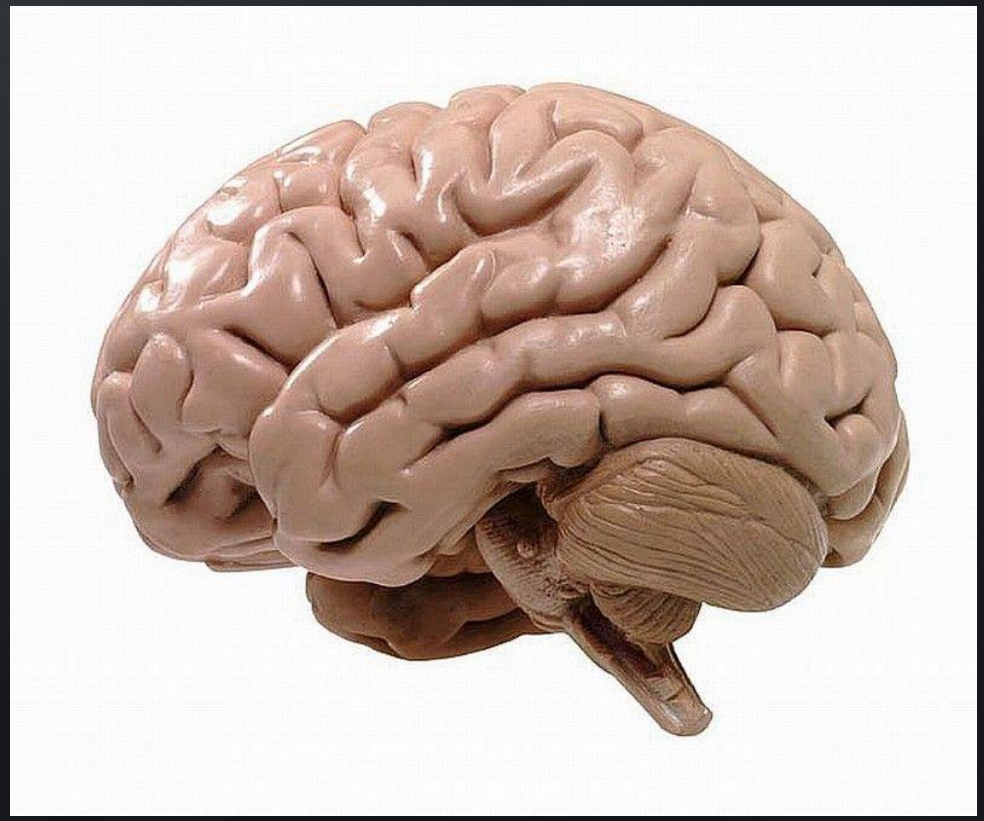
- Soft and fuzzy logic. Eventually consistent
- Soft contracts and dependencies
- Data duplication and polymorphing

Monolith vs Microservices



Monolith vs Microservices

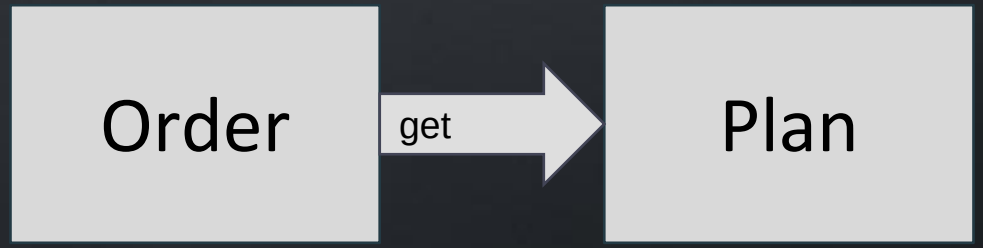




Monolith vs Microservices

Microservices are weak but they can adapt to circumstances

How to make components more independent

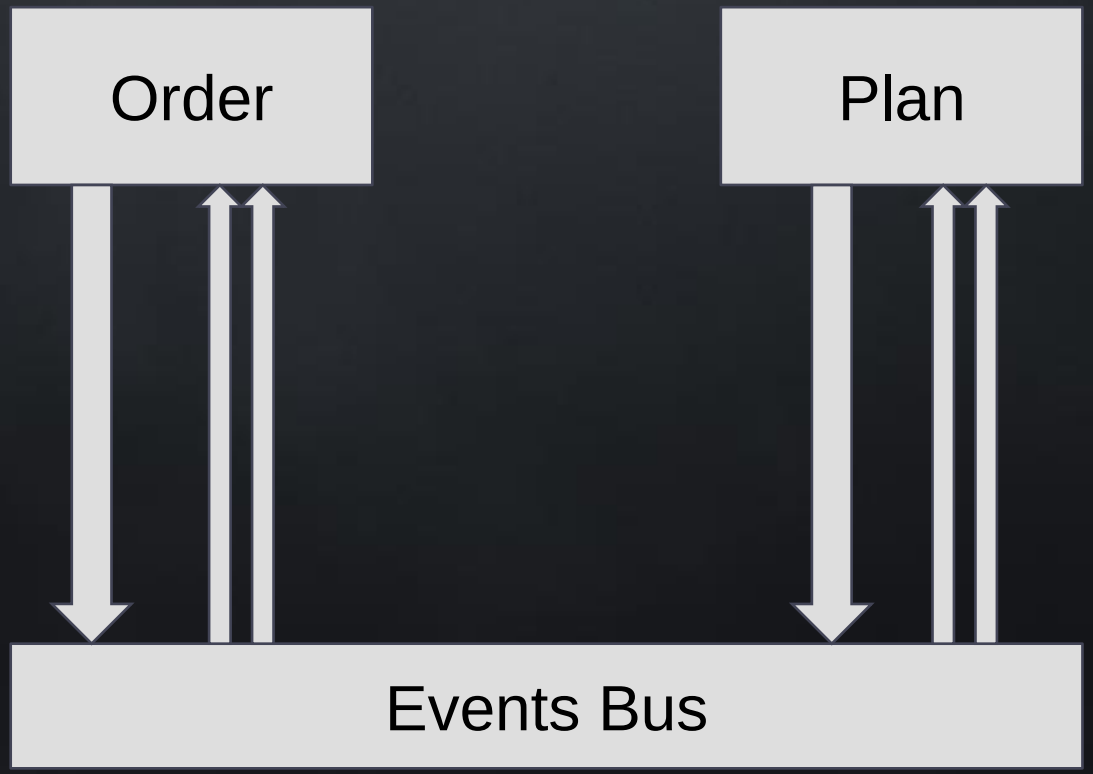


The event driven design paradigm

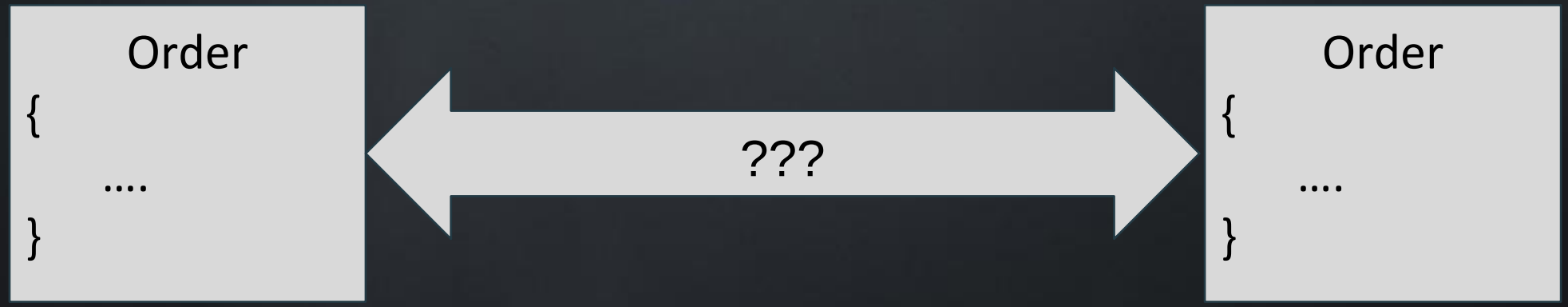
Martin Fowler

https://www.martinfowler.com/eaaDev/EventCollaboration.html

The event driven design for microservices



The problem of distributed storage

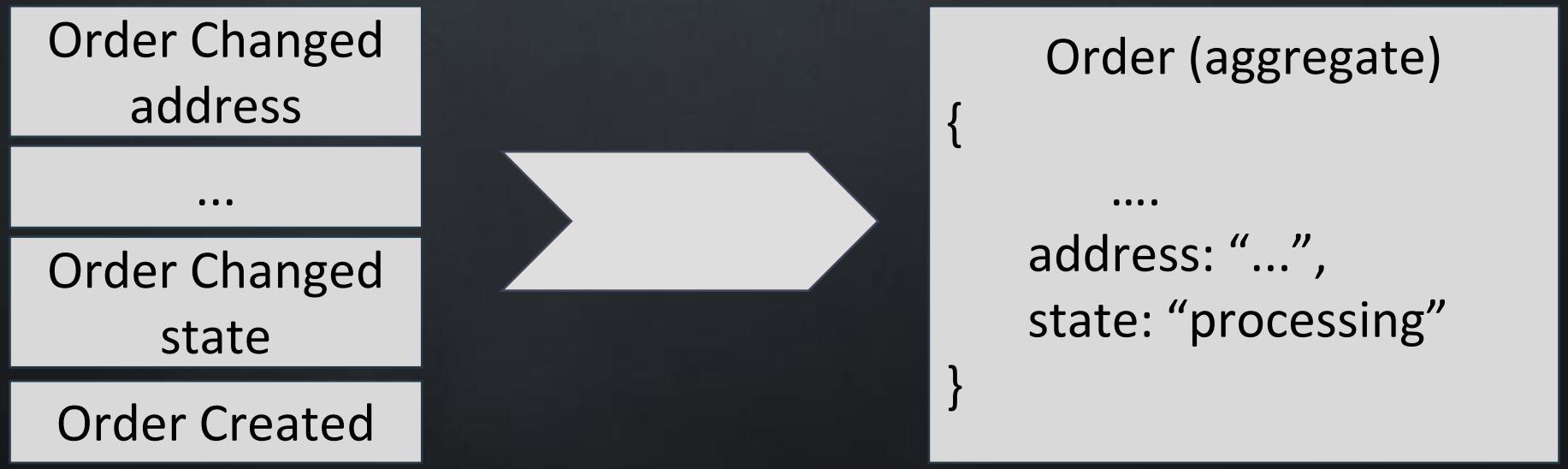


- We are used to store the final state of data
- Microservices are distributed system
- But how can we reconcile the data in the distributed system properly?

The problem of distributed storage

- Use the distributed transaction or 2PC (with locking and complexity)
- Use the event sourcing as a single source of true

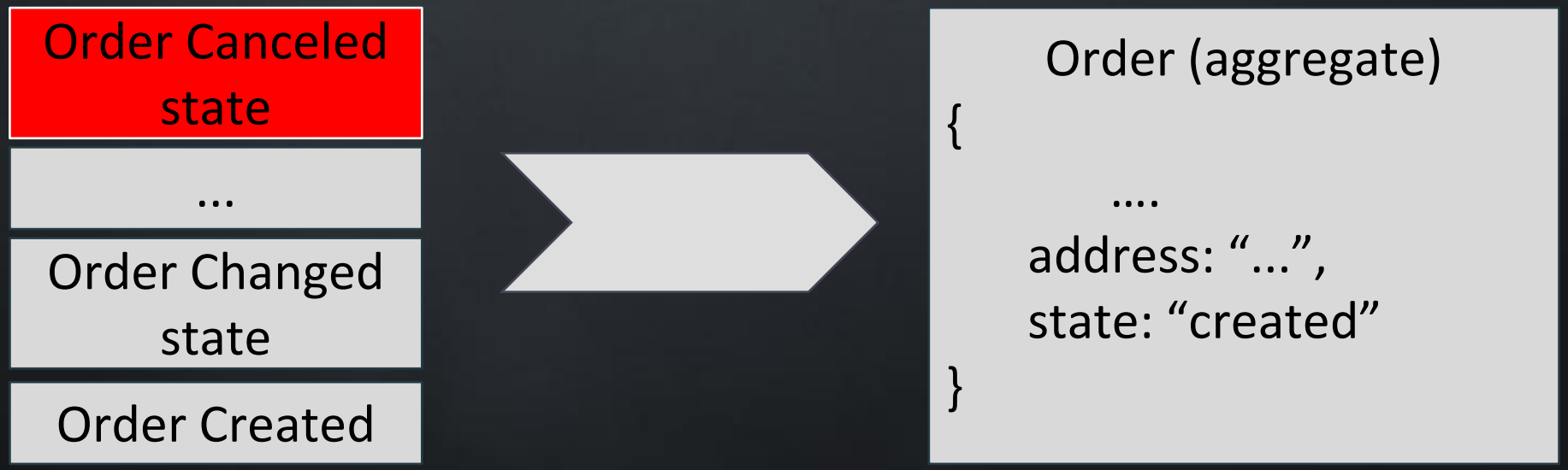
The event sourcing



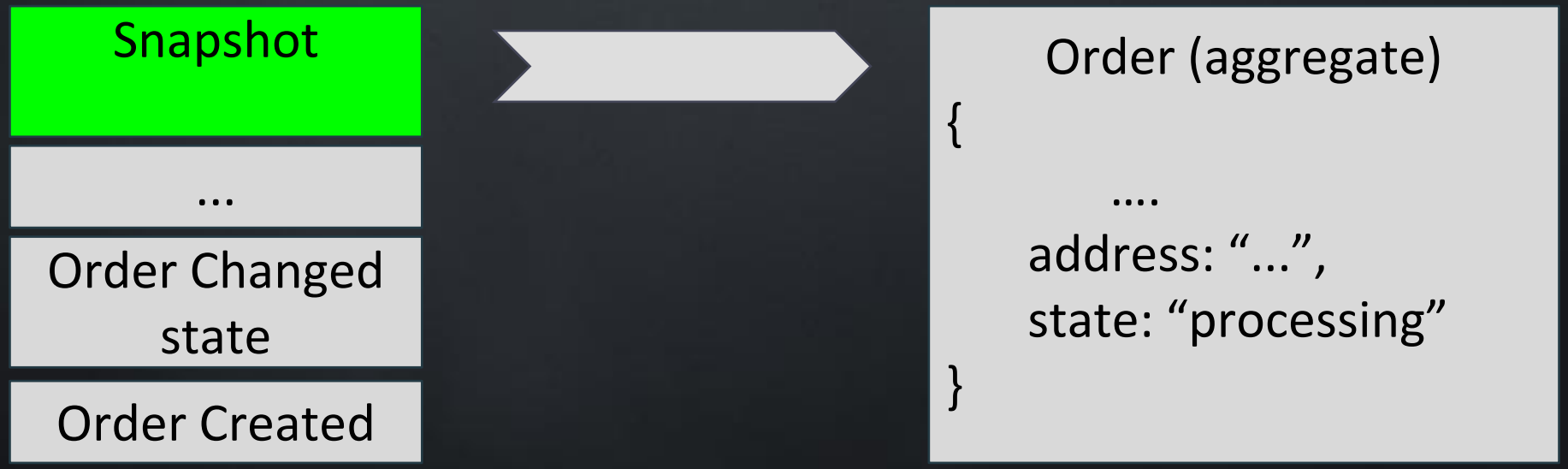
The event sourcing

- Single source of true
- Immutable
- Atomic
- Compensation events

The event sourcing



The event sourcing



The event sourcing

Event storage

- Event Log (file + index)
- DB Table + JSON
- NoSQL (Elastic, MongoDB)
- Kafka (with full persistence)

The event sourcing

- M. Fowler. https://martinfowler.com/eaaDev/EventSourcing.html
- Chris Richardson. https://microservices.io/patterns/data/event-sourcing.html
- Sebastian Daschner. https://blog.sebastian-daschner.com/entries/event_sourcing_cqrs_video_course
- Jonas Bonér. http://jonasboner.com/articles/

Eventual consistency

Or don’t worry about the Strong Consistency and just take the Eventual Consistency

- In distributed system the data can be inconsistent anyway (Frontend - Backend)
- If you use the event sourcing you can’t lose any data and all data will be consistent ultimately
- For critical transaction use the Soft Consistency(status changing or pending records)

The event bus as an environment

- Simple publisher-subscriber queue
- Capacity
- Persistency
- Performance
- Distributed

The event bus as an environment

- Kafka
- RabbitMQ (but it’s overuse)
- Amazon (SNS+SQS)
- NSQ

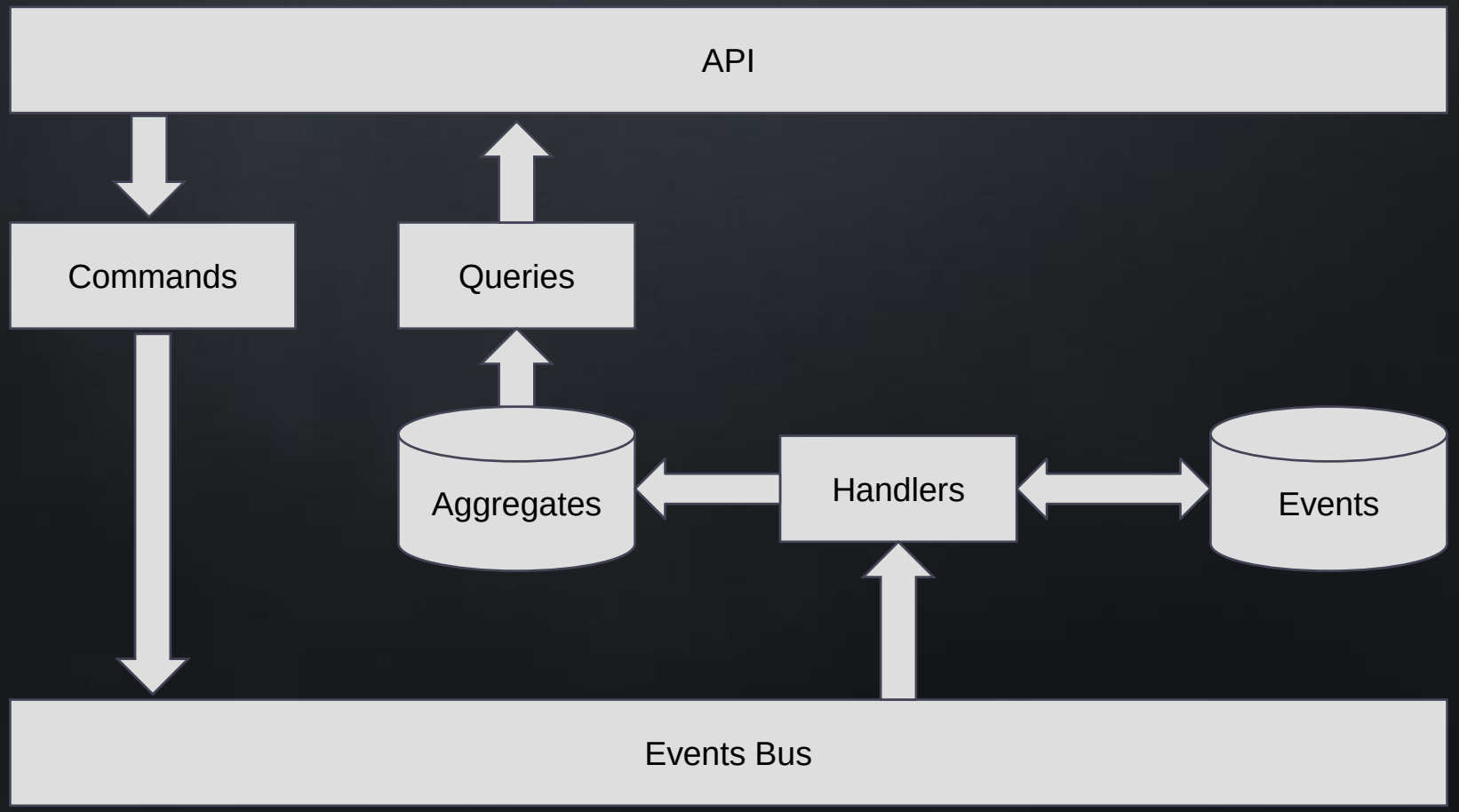
What do we have for agile microservices architecture creation?

- The event driven design paradigm
- The event sourcing storage
- The event bus as an environment
- The eventual consistency instead of transactions

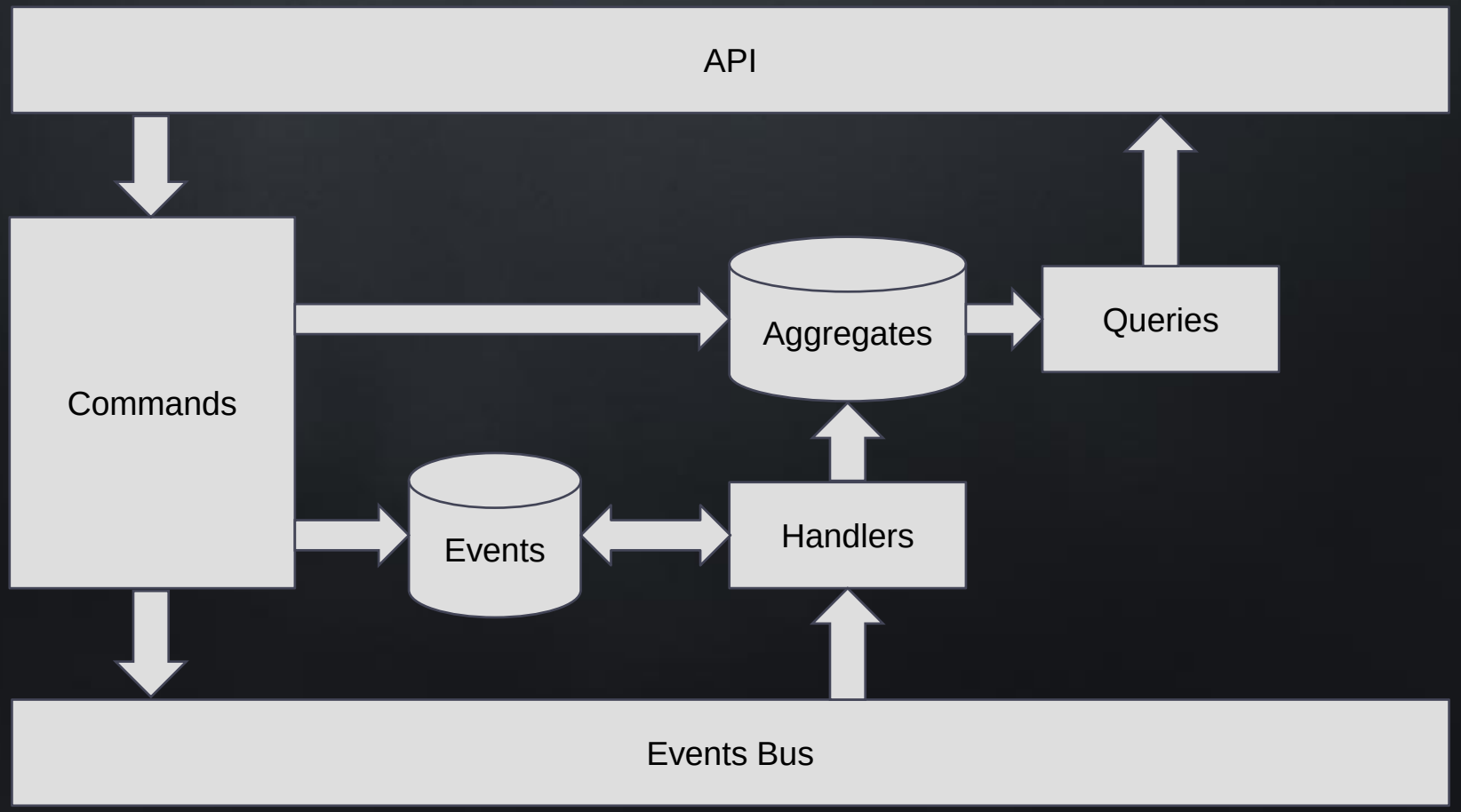
How to cook a microservice

- CQRS
- Stateless
- Resources oriented API
- Event handling

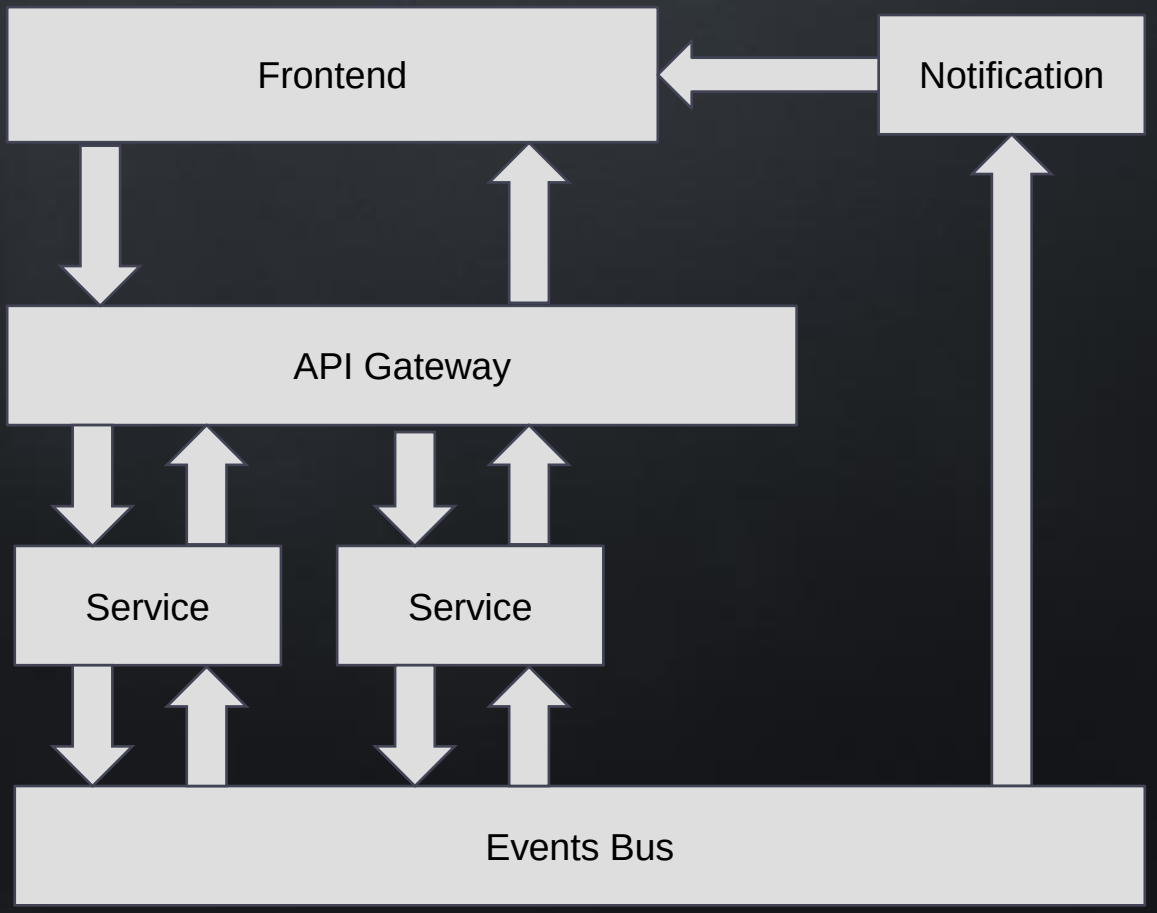
How to cook a microservice



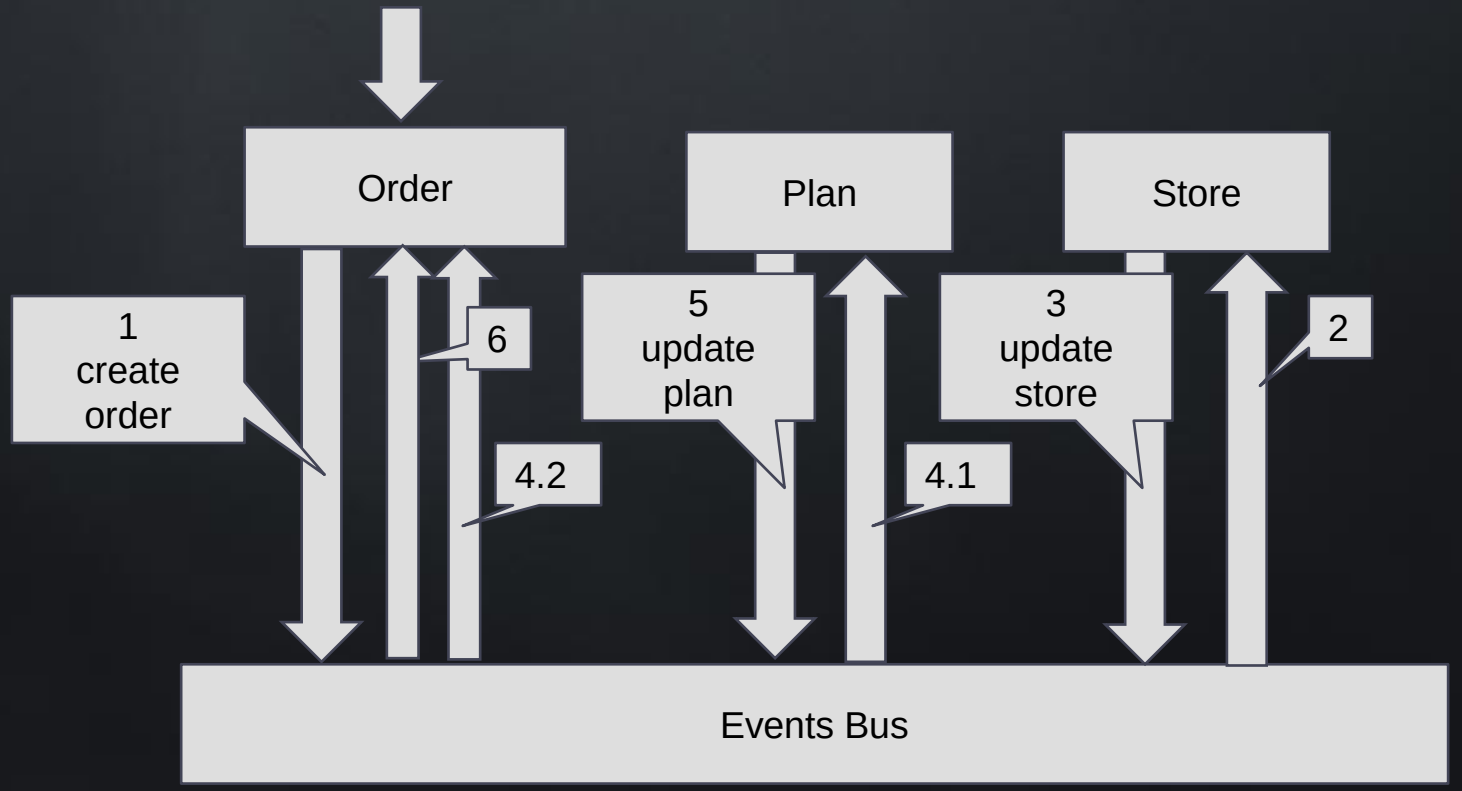
How to cook a microservice



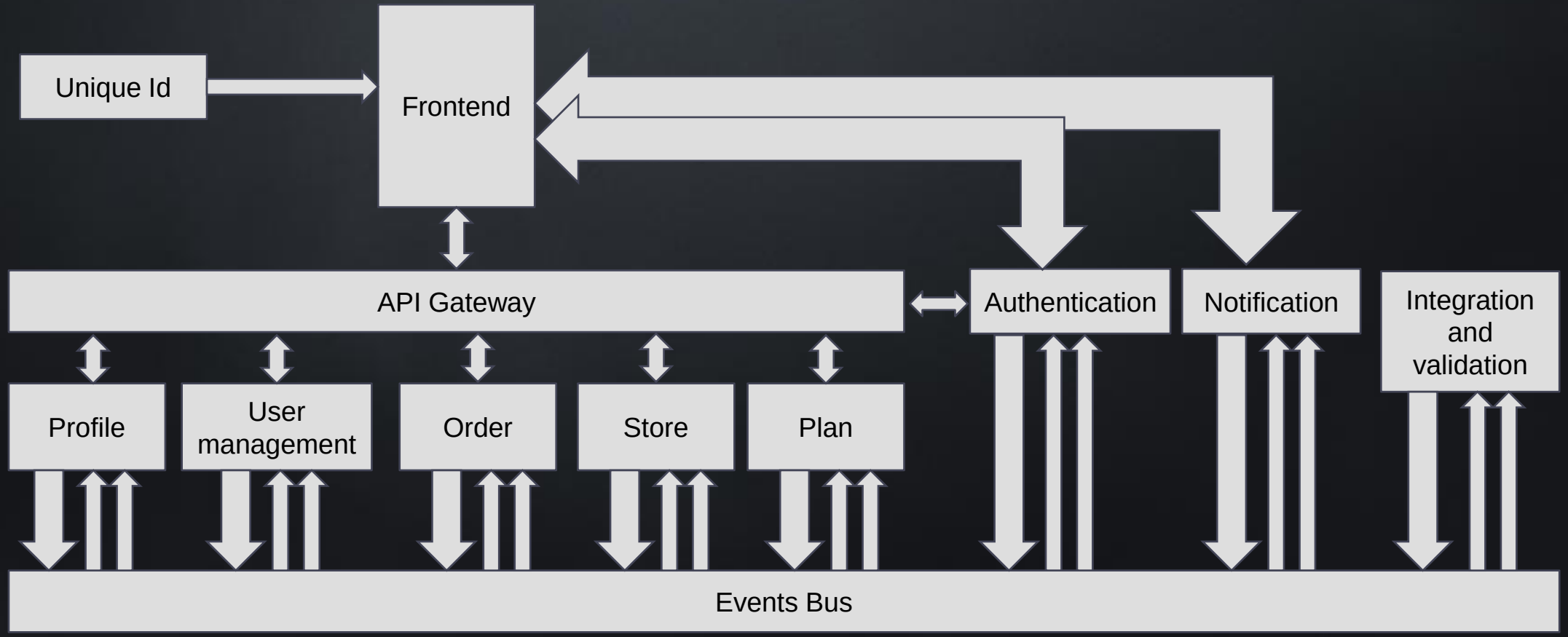
How to cook a microservice



Event driven microservices architecture



Event driven microservices architecture



Authentication and user management

- Use OAuth
- Frameworks: Spring, IndentyServer 4
- PaaS & SaaS: Auth0, Curity

https://oauth.net/code/

API Gateway

The Facade of your microservices

- Authentication
- Authorization
- Data aggregate
- Intelligent caching
- Discovering and balancing
- Frameworks: Spring, Ocelot
- SaaS & PaaS: Amazon, Oracle, APIGee
- Servers: Nginx, OSS: Tyk, DreamFactory, Kong, Traefik

Audit

- In the box since we have event sourcing
- ES for data aggregation

Integration service

- Data aggregation from different microservices
- Transfer data from one format to another
- Data adaptation
- Validation and data integrity
- Use the serverless lambdas
- SaaS & PaaS: Amazon, Google cloud functions
- OSS: OpenLambda, Serverless Framework, Kubeless

Unique ID

- Don’t use auto-increment for ID at all
- Don’t generate any ID in the domain services
- Use UUIDs
- Use sequences
- You can your own implementation or any based on Twitter's Snowflake

DevOps

- Full team member
- Full time
- The right hand of architect

Summary

- The microservices come to us for a long period
- The whole team need to realize the key benefit of the microservices architecture first: flexibility and adaptation
- Use best practices like as the event-driven approach, the event sourcing and CQRS, existing services and frameworks
- Don’t miss DevOps!

Contact Me

- Alexandr Shcherbakov
- Email: alexandr.sherby@gmail.com
- https://auriga.com/