

# MICROSERVICES - IN DEPTH

**Expert Connect Session** 

Speaker



Chandra Mouli

Solution Architect







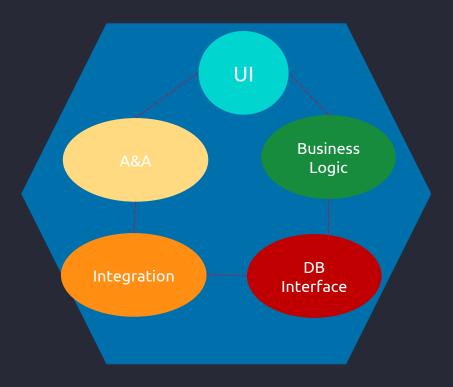
- Recap
  - Overview of Microservices Architecture & Challenges
  - Microservices Patterns
- Microservices main characteristics
- Service infrastructure principles
- Q&A •
- Think out of the box

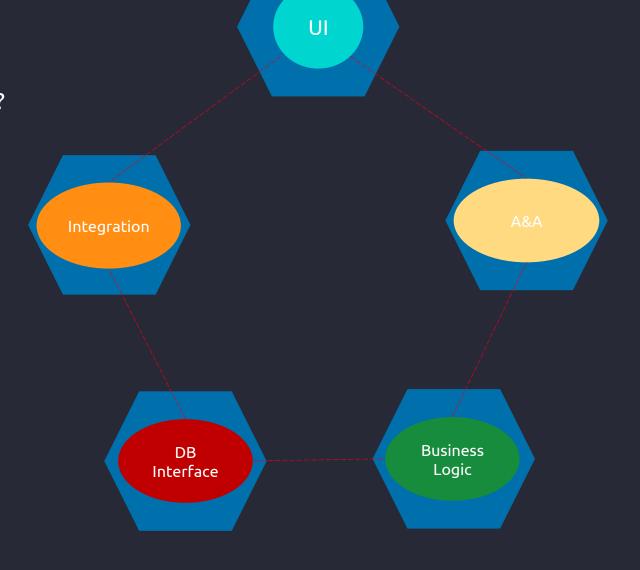
# WHO AM I?

- 20+ years of IT Experience
- Role <u>Solution Architect</u>
- Expertise in
  - System Design
  - Java & J2EE Development
  - Cloud
  - Integration

### **OVERVIEW OF MICROSERVICES - RECAP**

- What is Microservice Architecture?
- When should we choose microservice architecture?
- Challenges in Microservices



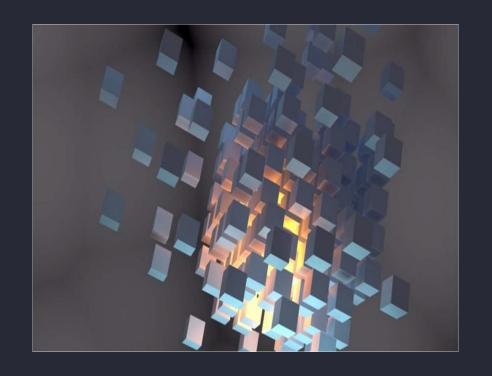




#### MICROSERVICES MAIN CHARACTERISTICS

#### A 'good' microservice have 7 main characteristics

- 1. Fine-grained capabilities that are stateless Eg:A&A
- 2. Well <u>defined interfaces</u> which 'hide' how the service is executed. Eg: Forex rate & Collect payment from different bank
- 3. A 'very loosely-coupled' approach (can change one service without impacting another)
- 4. Autonomous, explicitly versioned & completely independent
- 5. stupid "do one thing, and do it well"
- 6. cost and value fully defined
- 7. Disposable

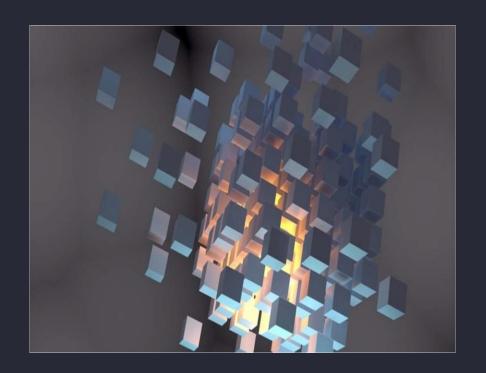




#### SERVICE INFRASTRUCTURE PRINCIPLES

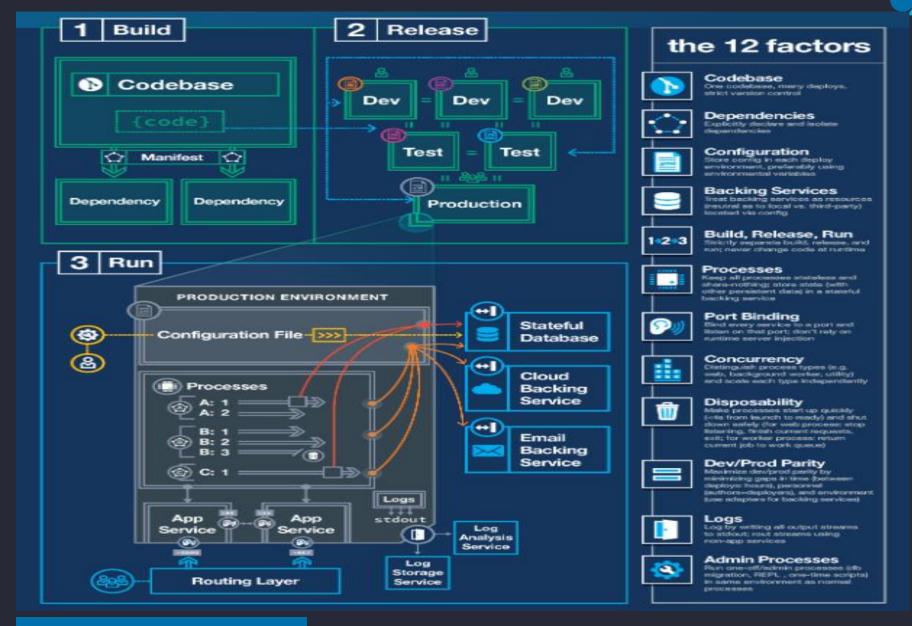
#### A 'good' microservice have 7 main characteristics

- 1. Utility based
- 2. Open standards based
- 3. Multi-Tenant support
- 4. Technology independent & virtualized
- 5. Service oriented
- 6. Scalable
- 7. Fit-for-purpose
- 8. Sustainable
- 9. Automated
- 10. Modularity



#### 12-FACTOR APP

Java community Expert session | 26-Oct-2023





### SOME MORE DESIGN PATTERNS

Decomposition **Patterns** 

> Decompose by **Business Capability**

Decompose by Subdomain

Decompose by **Transactions** 

Strangler Pattern

Bulkhead Pattern

Sidecar Pattern

Integration **Patterns** 

**API Gateway Pattern** 

**Aggregator Pattern** 

Proxy Pattern

**Gateway Routing** Pattern

Chained Microservices Pattern

Branch Pattern

Client-side UI Composition Pattern Database **Patterns** 

> Database per service pattern

Shared Database per service pattern

**CQRS** pattern

**Event Sourcing** pattern

Saga pattern

Covered

Not -Covered Today session Observability **Patterns** 

> Log aggregation pattern

Performance metrics pattern

Distributed tracing pattern

Health check pattern

Cross-cutting Concerns Patters

> External configuration pattern

> > Service discovery pattern

Circuit breaker pattern

Blue-Green deployment pattern

Canary deployment pattern

> Dark Launching pattern

A/B testing pattern

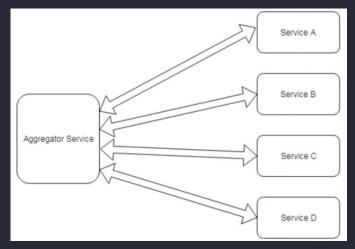
https://microservices.io/patterns/ https://dennylesmana.medium.com/microservices-deployment-patterns-ca1343c89e13



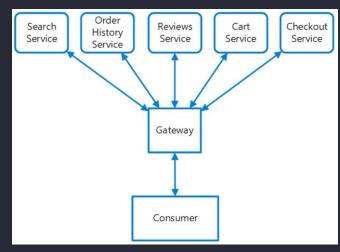
# SOME MORE DESIGN PATTERNS

#### **API** Gateway Features

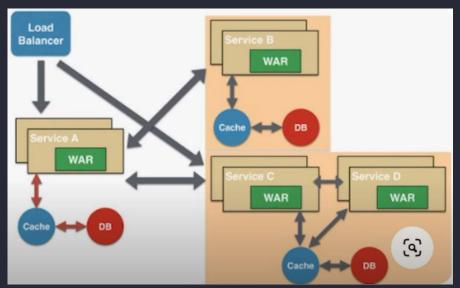
- Authentication and authorisation
- Service discovery integration
- Response caching
- SSL Termination
- Retry policies, circuit breaker, and Quality-of-Service
- Rate limiting and throttling
- o Logging, tracing, correlation
- Headers, guery strings, and claims transformation

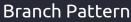


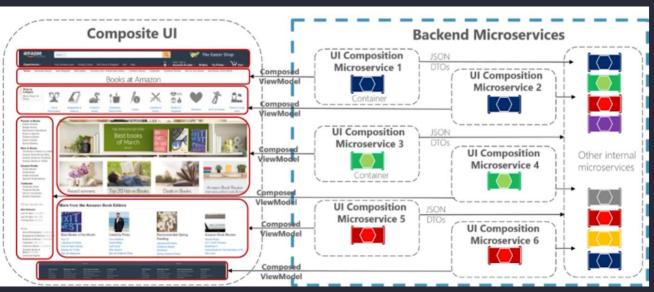
Aggregator Pattern



Gateway routing Pattern



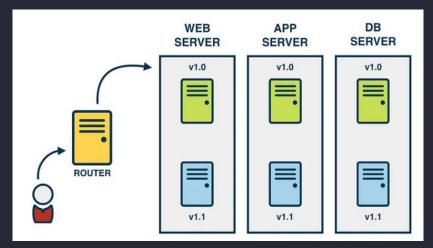




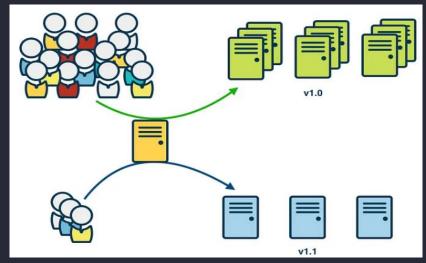
Client-side UI Composition Pattern



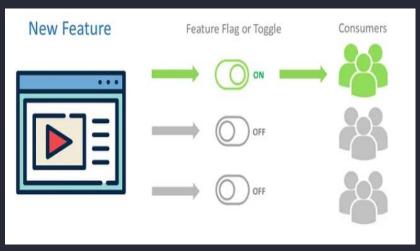
# SOME MORE DESIGN PATTERNS



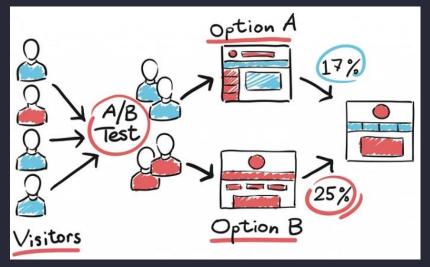
Blue/Green deployment



Canary Deployment



Dark Launching

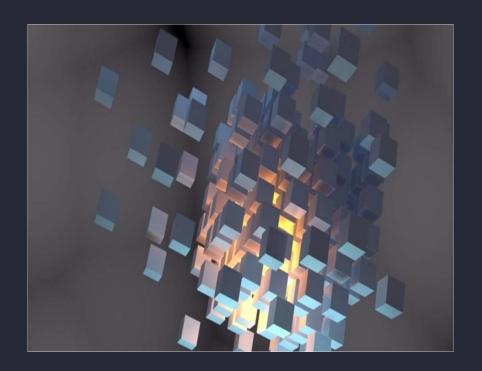


A/B Testing



### **CHALLENGES**

- 1. Logging.
- 2. Monitoring.
- 3. Service impact (Caused By One Service, Can Cause Trouble Elsewhere)
- 4. Version management & Cyclic dependency
- 5. Container Security
- .. more





Q & A



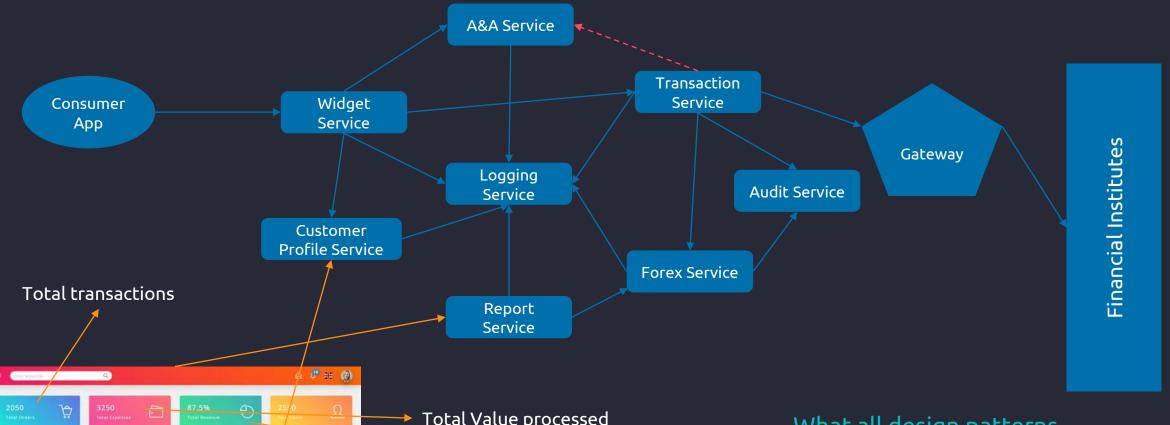
# THINK OUT OF THE BOX



• Microservices based framework to provide basic CRUD operation (Plug and Play)

### THINK OUT OF THE BOX





Total Value processed

Heat map

Successful transactions

Application wise transactions

What all design patterns implemented in Simple Payment processing with Audit tracing and widget style implementation?



