EXPLORING SERVERLESS DESIGN PATTERNS

Design and Develop a Serverless Event-Driven Microservice-Based Solution

Overview of Serverless Design Patterns

Understanding the Importance and Characteristics of Serverless Design Patterns

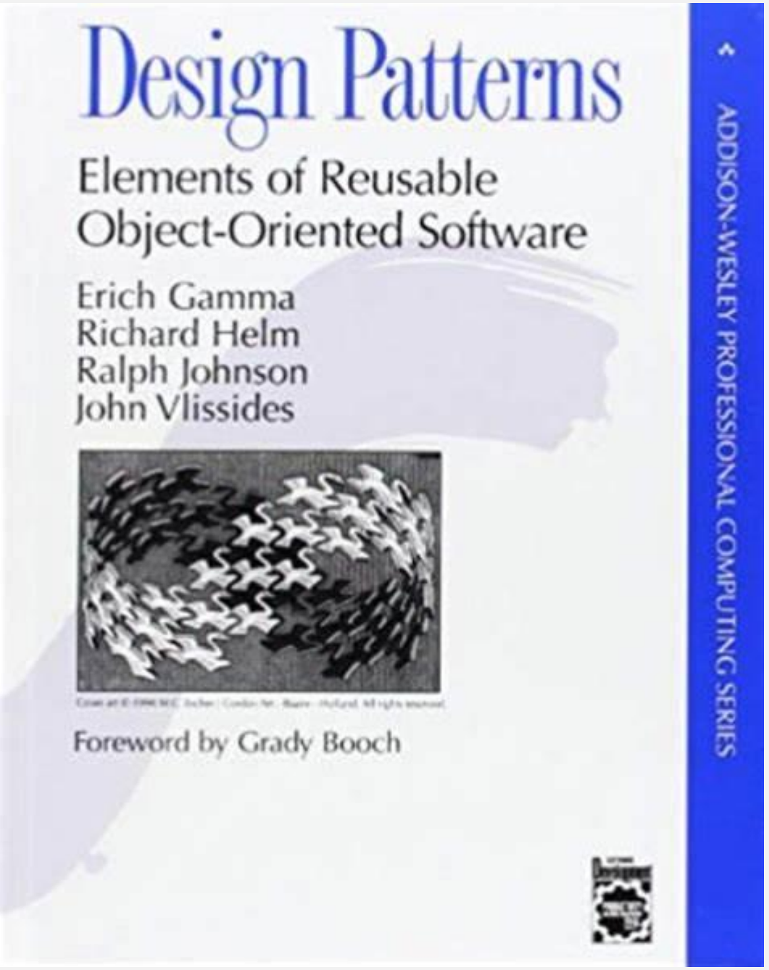
What are design Patterns

- Reusable solutions to common problems
- Best practices and proven solutions
- Building blocks for maintainable, scalable, and robust software

Why Design Patterns Mattern

- Address complexity
- Encourage best practices and standardization
- Enhance code readability and maintainability
- Facilitate collaboration

Gang of Four



Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

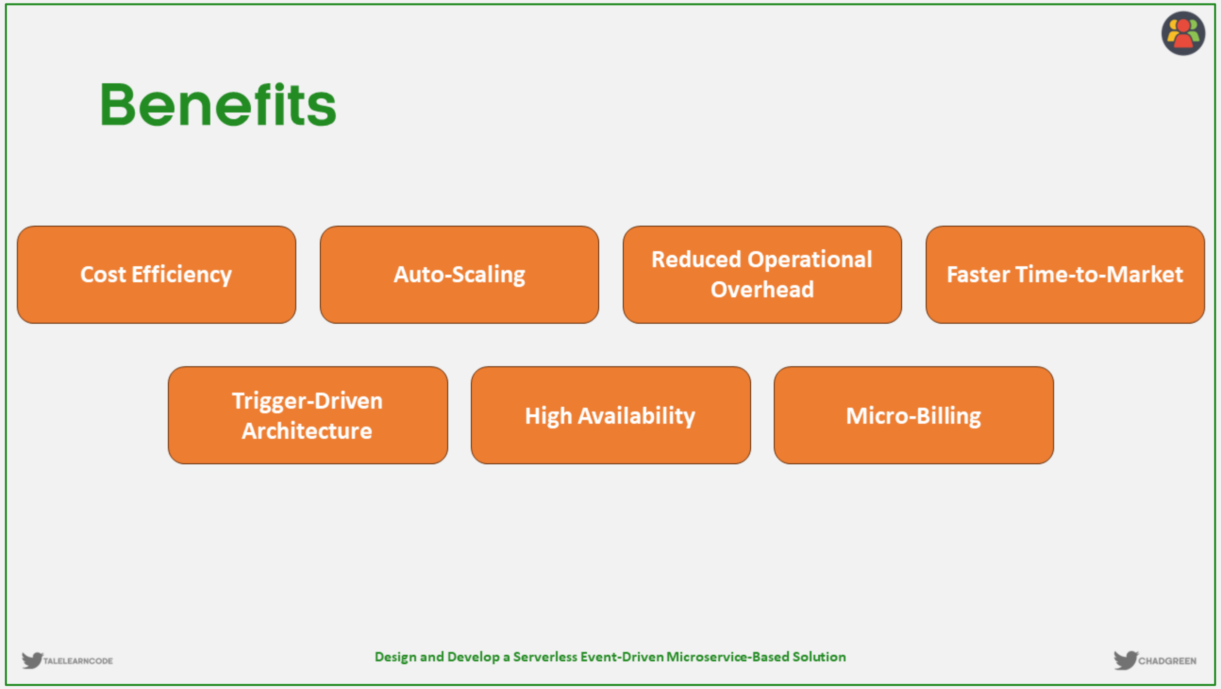
Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Types of Design Patterns

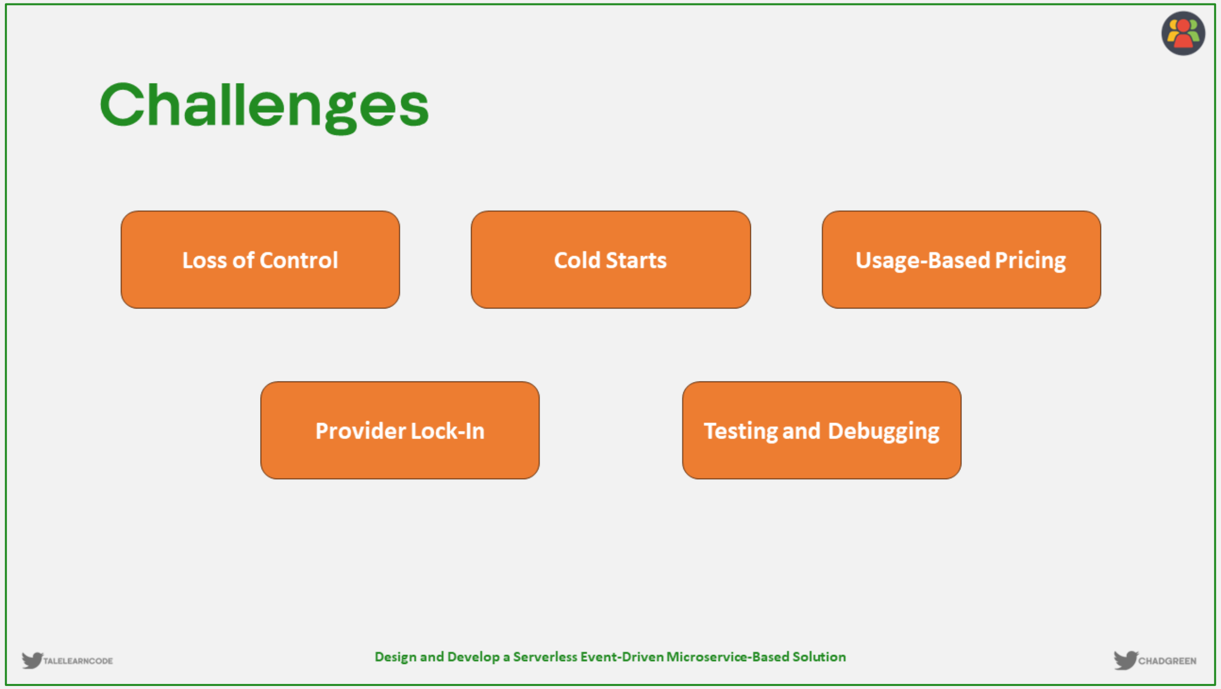
- Creational
- Structural
- Behavioral
- Concurrency
- Architectural
- Cloud

Serverless Design Patterns



Benefits

- Cost Efficiency
- Auto-Scaling
- Reduced Operational Overhead
- Faster Time-to-Market
- Trigger-Driven Architecture
- High Availability
- Micro-Billing



Challenges

- Loss of Control
- Cold Starts
- Usage-Based Pricing
- Provider Lock-In
- Testing and Debugging

Common Characteristics

Event-Driven

Scalability

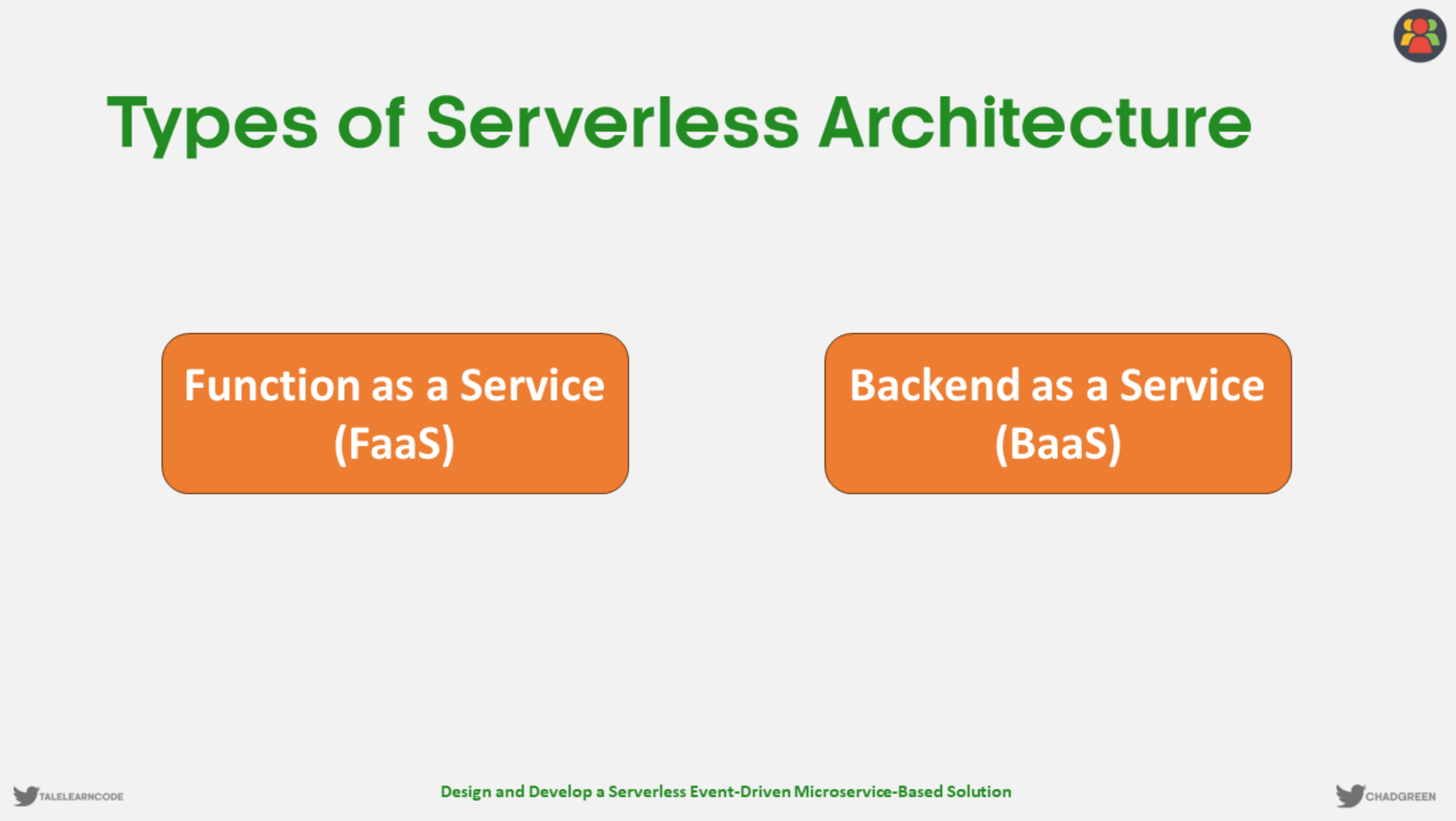
Microservices

Managed Services

Exploring Serverless Design Patterns

Sample of Serverless Design Patterns

High-Level Patterns

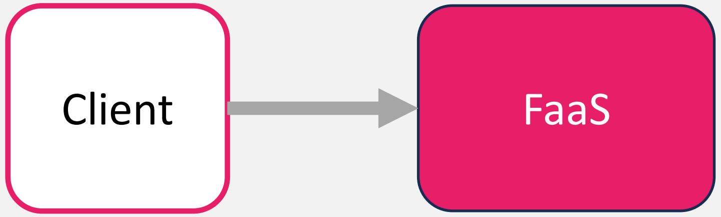


Types of Serverless Architecture

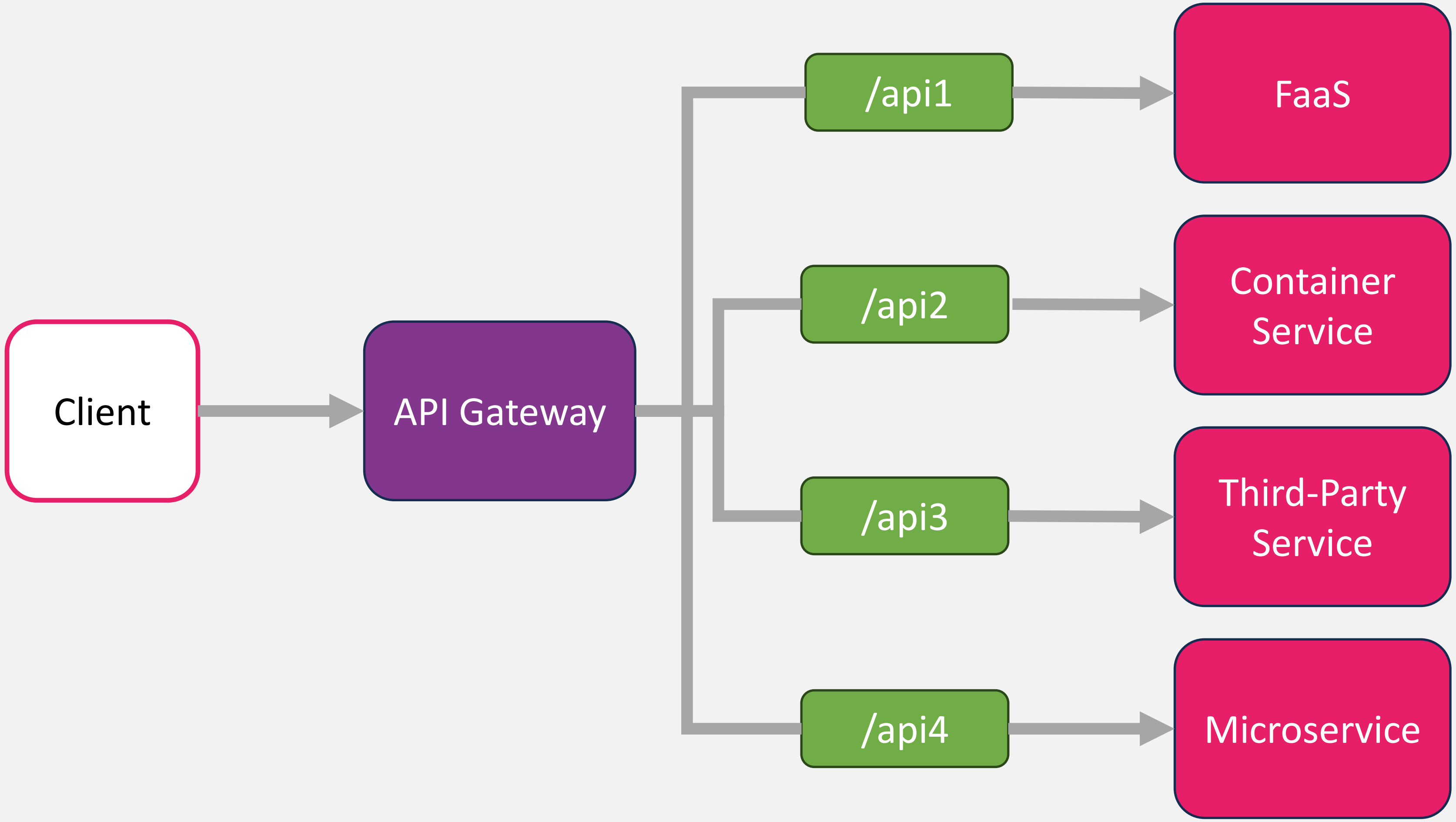
Function as a Service (FaaS)

Backend as a Service (BaaS)

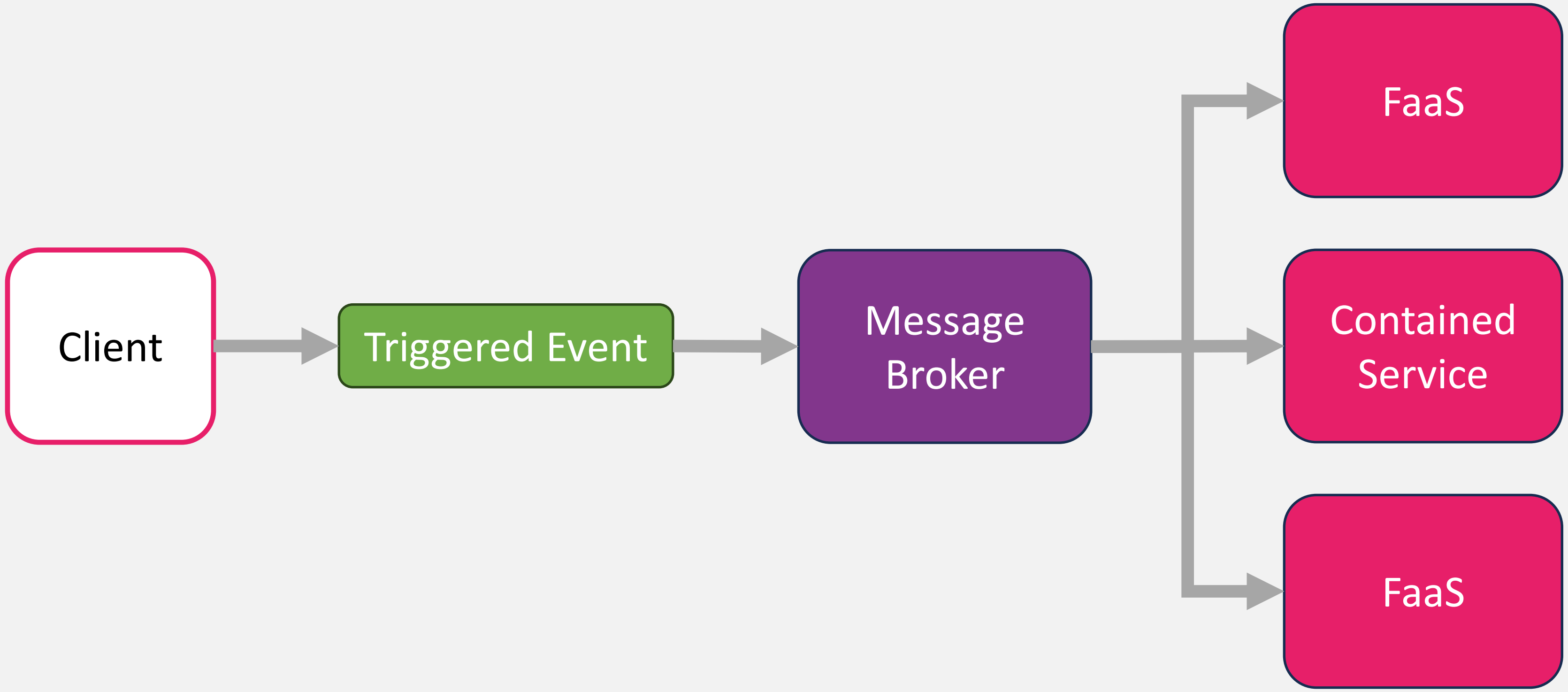
Simple Web Service



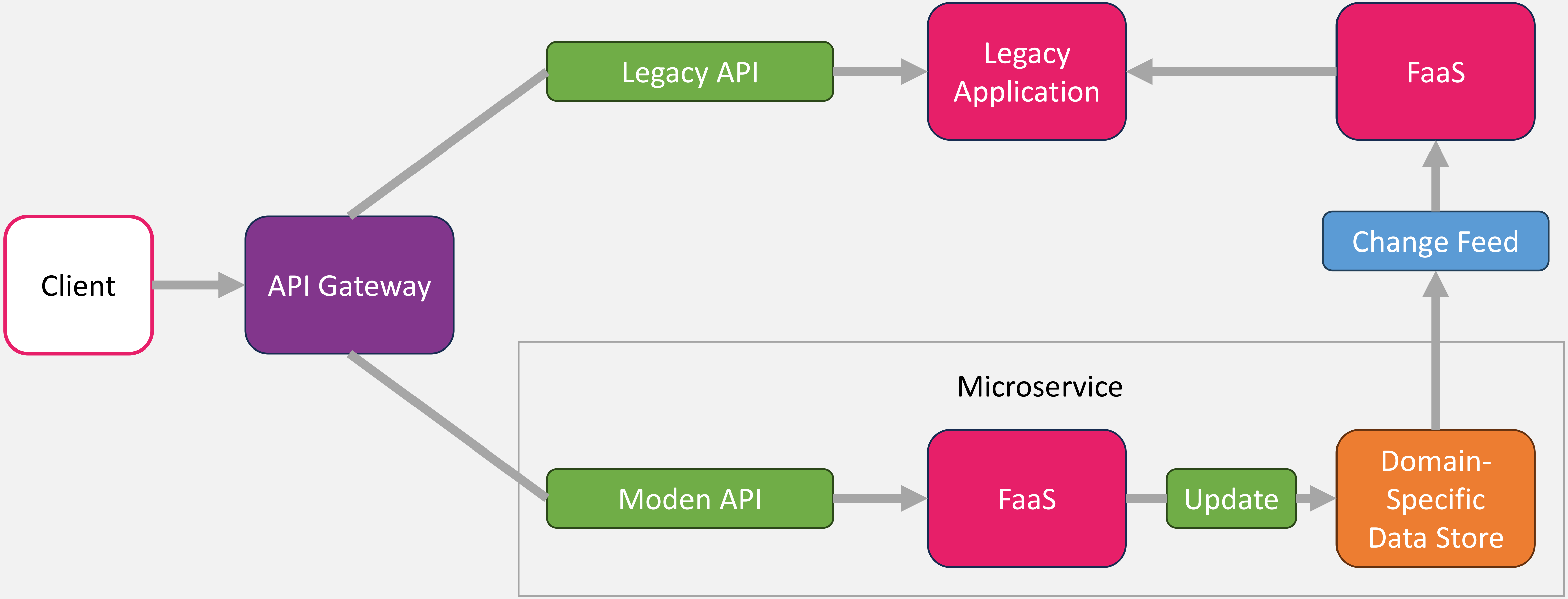
API Gateway


Client
API Gateway
/api1
/api2
/api3
/api4
FaaS
Container Service
Third-Party Service
Microservice

Decoupled Messaging



Strangler


Legacy API
Legacy Application
FaaS
Client
API Gateway
Change Feed
Microservice
Moden API
FaaS
Update
Domain-Specific Data Store

Learn More

Thursday – 4:00 pm
Lancaster 2&3

Friday – 11:30 am
Arbor 2

BREAK

Back at 10:45

LAB SCENARIO