Building Scalable and Secure APIs

Amanda Hua

Agenda

Evolution of API

API Design Principles

API Scalability

API Security

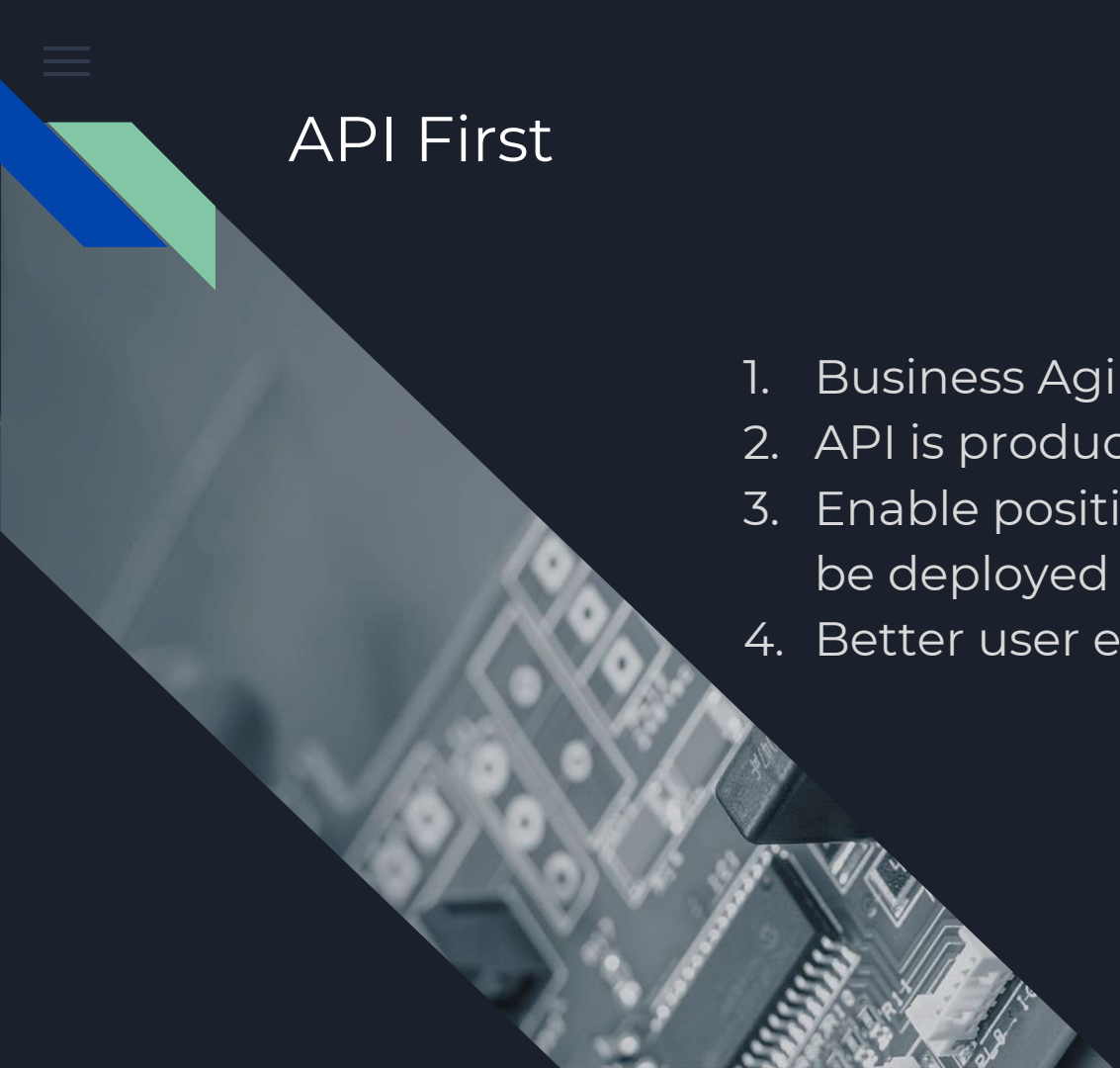
API Backward Compatibility

Q&A

What’s API?

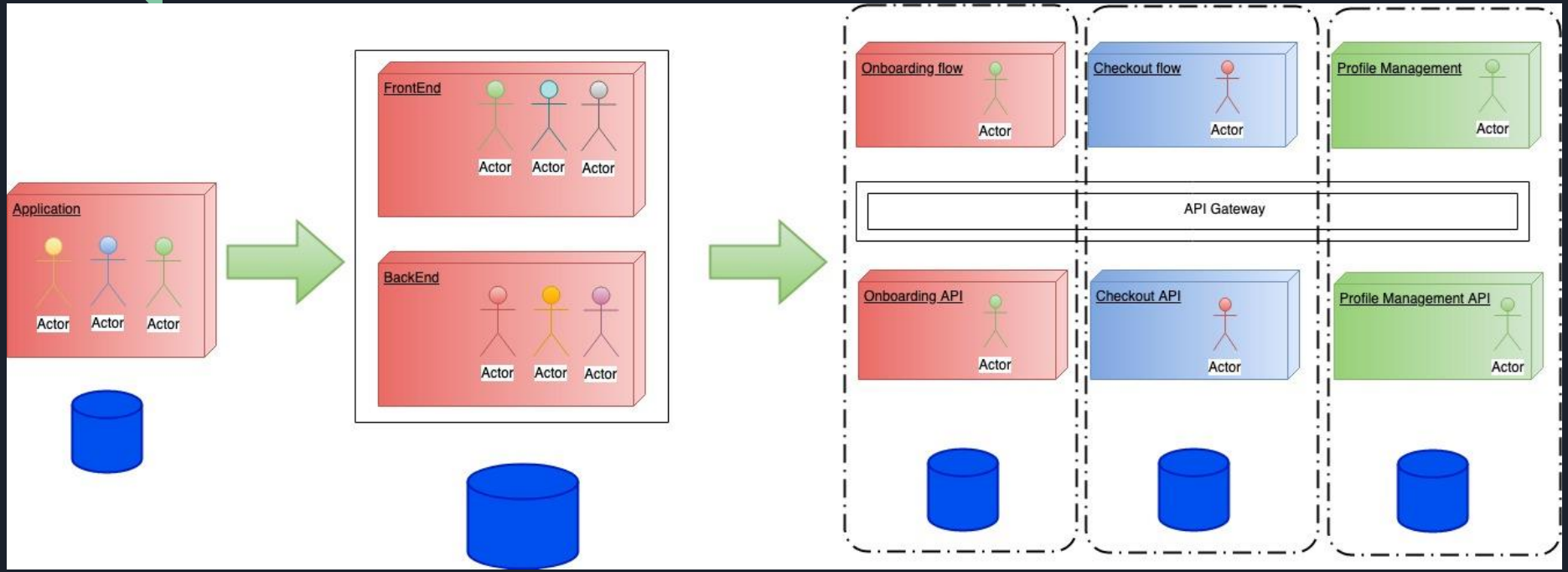
API stands for Application Programming Interface

API First

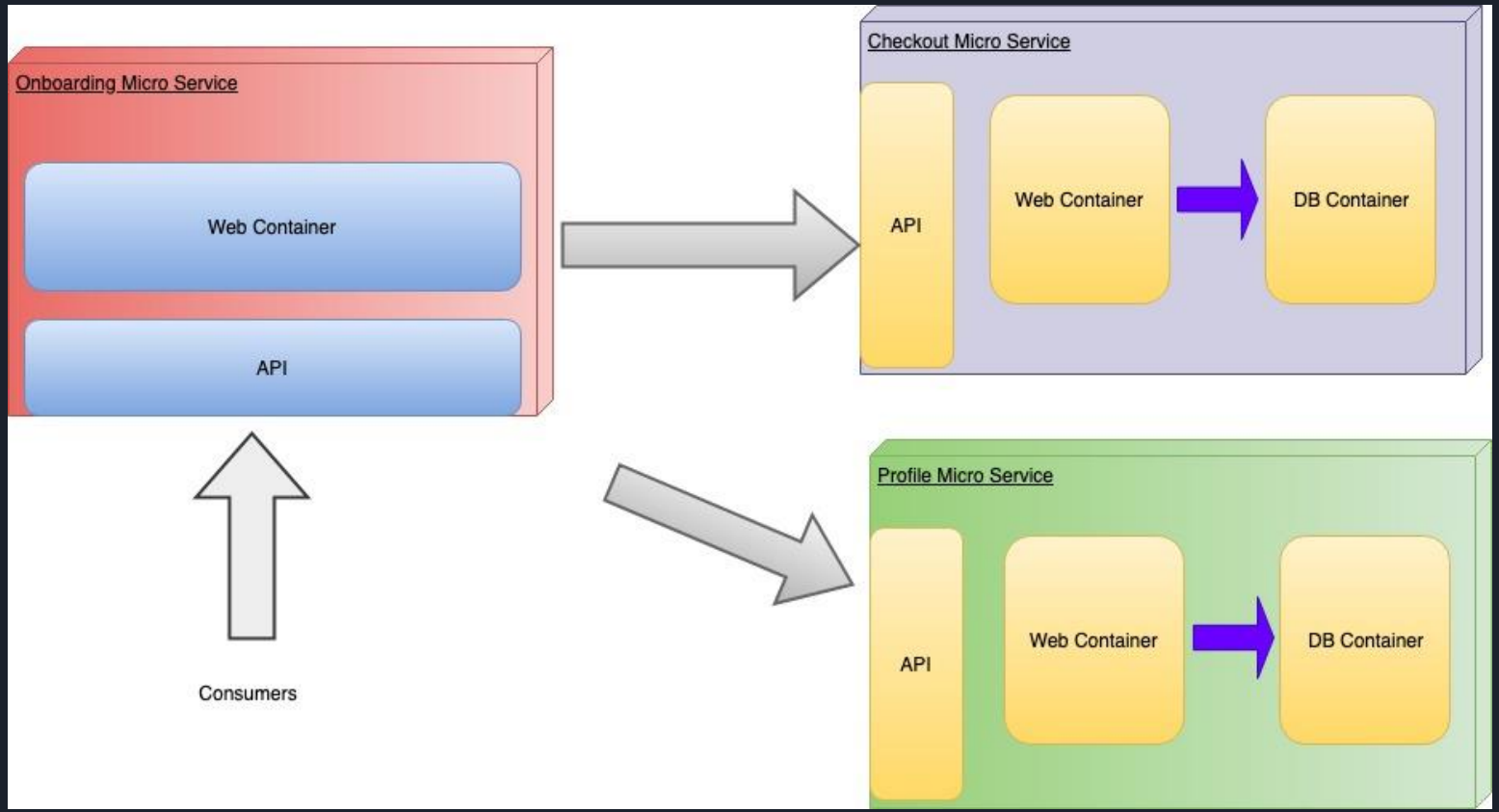


1. Business Agility
2. API is product
3. Enable positive ROI (quick wins and be deployed across multiple areas)
4. Better user experience

Evolutions of API



A Common Architecture of Microservices



API Design Principles and Standards (I)

01 Decoupling

02 Share and Re-use

03 Encapsulation

04 Consistency

05 Easy of Use

06 Stability

07 Data Protection

API Design Principles and Standards (II)

08 Versioning Strategy and Backward Compatibility

09 API Analytics and Usage Report

10 Governance

11 Documentation

What Does Scalable Mean?

01 Extensible

02 Built into the architecture

03 Implies demand balancing: Scalability implies that the handling of traffic is just done as well with one hundred requests as with one million

API Scalability Design Guideline

01 Stateless session

02 Lightweight design

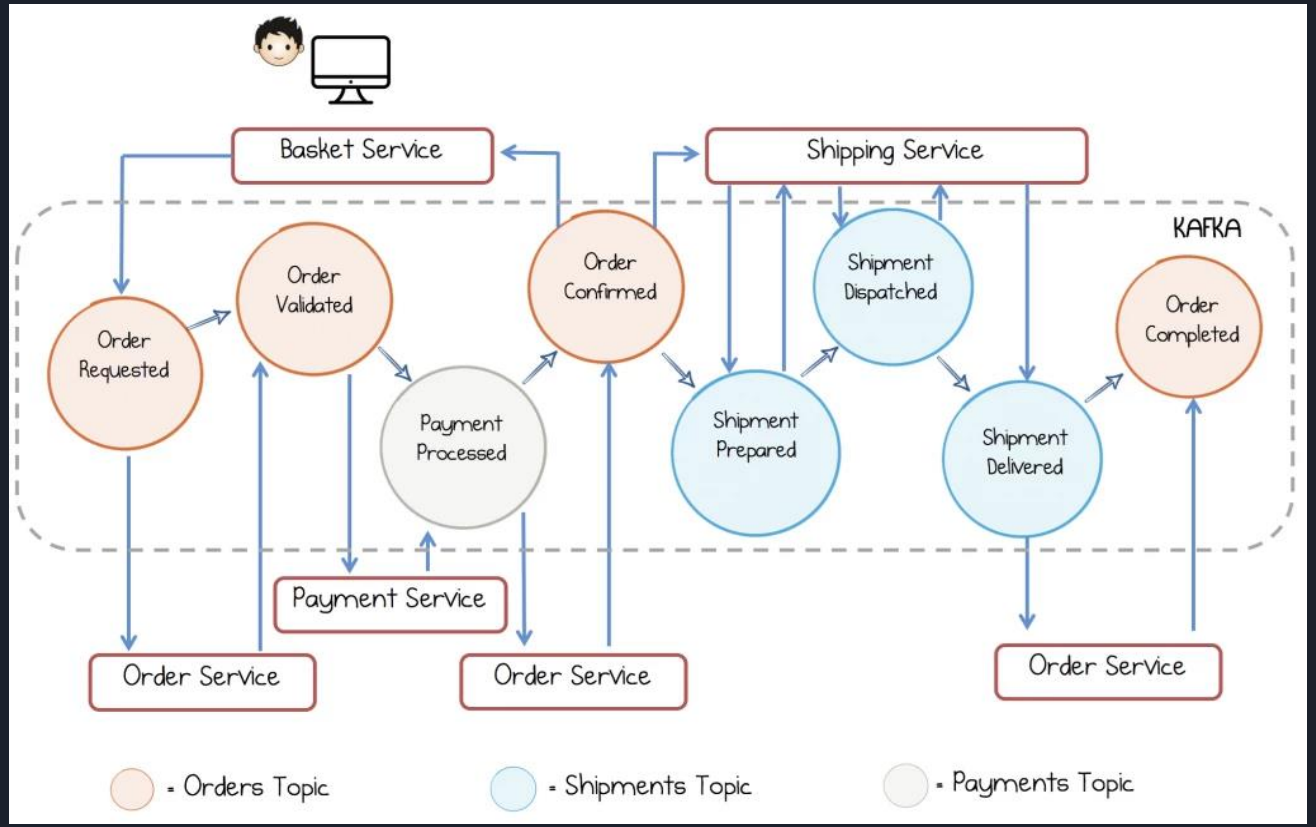
03 Resource pooling

04 Adoption proven patterns

05 Optimal enterprise integrations

06 Testing, testing, testing

API Scalability: Decoupling



API Scalability: Use a common API layer to ground the cloud

Create an integration layer that connects solutions within the firewall and on the outside and takes care of API impedance mismatches and error handling

API Scalability: Coordinate Business and Technical Monitoring

01 Need to integrate multiple data sources (internal and external) to gain clarification and insights

02 It’s imperative to construct meaningful links between systems and demonstrate the value of API initiatives

03 Key to gain end-to-end comprehensive business and technical metrics and visibility

API Scalability: Ensure a Positive User Experience and Satisfy SLAs

01 Keep iterating and set the expected service levels for services

02 Use predictive analytics to give business and technical users ability to monitor, predict and proactively act on changes in performance

API Scalability: Build Governance as you need it

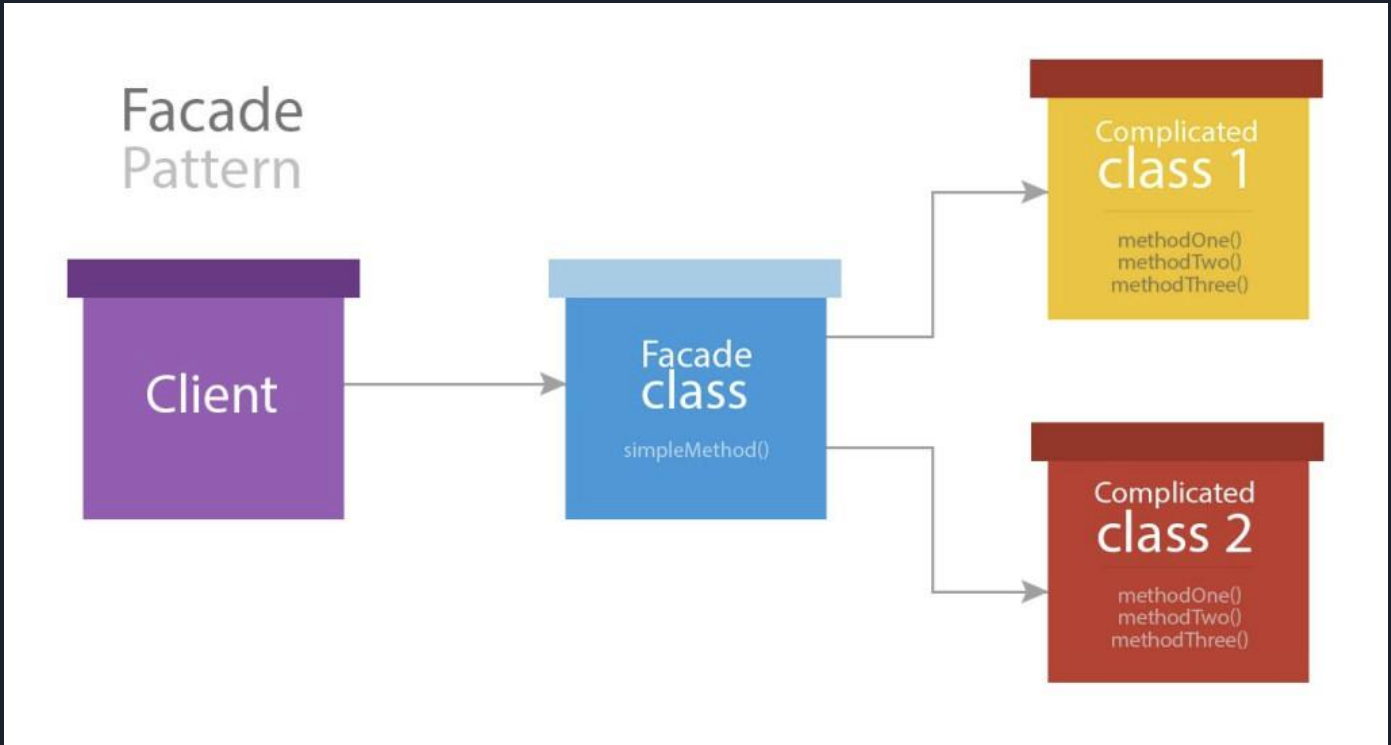
01 Consistently enforcing policies and contracts

02 Shielding consumers from changes

API Scalability: Leverage Infrastructure

	Nginx	Kong	Tyk[2]	Ambassador	AWS API Gw
Basic[1]	✓	✓	✓	✓	✓
OAuth	✓	✓	✓	✓	✓
JWT	✓	✓	✓	✓	✓
Ip Listing	✓	✓	✓	✗	✓
Analytics	✓	✓	✓	✓	✓
Rate Limiting	✓	✓	✓	✓	✓
Transformation	✓	✓	✓	✓	✓
gRPC	✓	✓	✓	✓	✗
Websockets	✓	✓	✓	✓	✓
Service Mesh	✗	✓	✓	✓ 4	✗
Kubernetes Ingress	✓	✓	✓	✓	✗
Caching	✓	✓	✓	✗	✓
Documentation	✓	✓	✓	✗	✓
Admin UI	✓	✓ 3	✓	✗	✓
Ease of Setup	3.5	4	2.5	3	5

API Security - API Facade Pattern



API Security - Layers and In-depth

01 Role-based access control

02 Refresh credentials

03 Pen testing

04 Security testing

05 Security by design

API Security - Access Control

01 Authentication and authorization

02 User throttling and quota management

03 Separate policies from API

04 Auditing

API Security - Design Guideline (I)

01 Data collections and securities

02 Meet compliance requirements

03 Good balance with UX and performance

04 Security by obscurity

05 Weakest link

07 Defense in depth

08 Insider attacks

API Security - Design Guideline (II)

01 Lowest privilege

02 Fail-safe defaults

03 Complete audit trails

04 Least common mechanism

API Security - Design Guideline (III)

01 Confidentiality

02 Integrity

03 Availability

04 Users, apps

05 Clients, servers

API Security - Real-time Analytics

01 Provide in-depth insights from both data-in-motion and data-at-rest

02 Proactively identify API issues before business impact

03 Quickly diagnose root causes

04 Security dashboard

Why Backward Compatibility?

01 From business perspective, lost trust from customers

02 From product perspective, it’s hard to sustain

03 From engineering perspective, incompatibility causes NP-Complete problems

04 Key is to balance the strictness of BC (99%)

Recap

01 Design at scale

02 Infrastructure

03 Kafka, cache, data

04 Usage report, monitor/alert

05 Think big, start small and keep iterating

Thank you!

Q&A