API Workshop

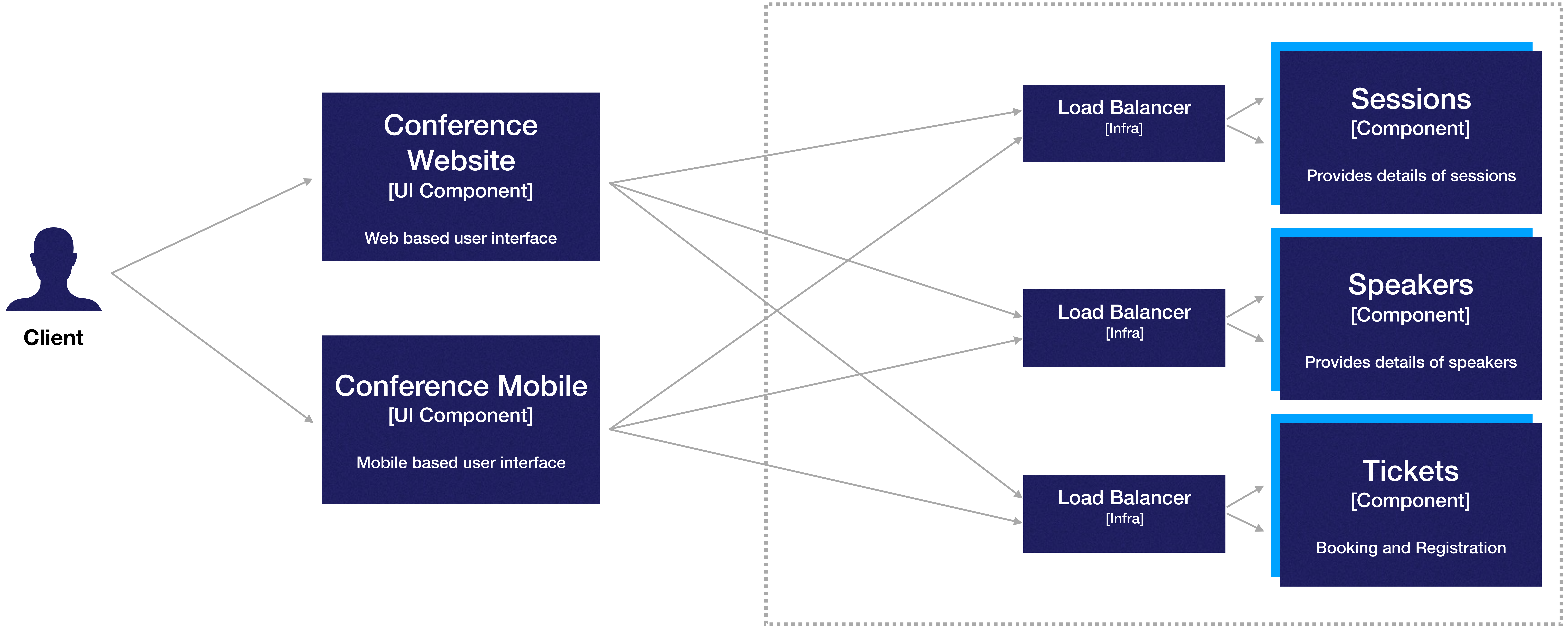
Gateways

O'REILLY®
Software Architecture

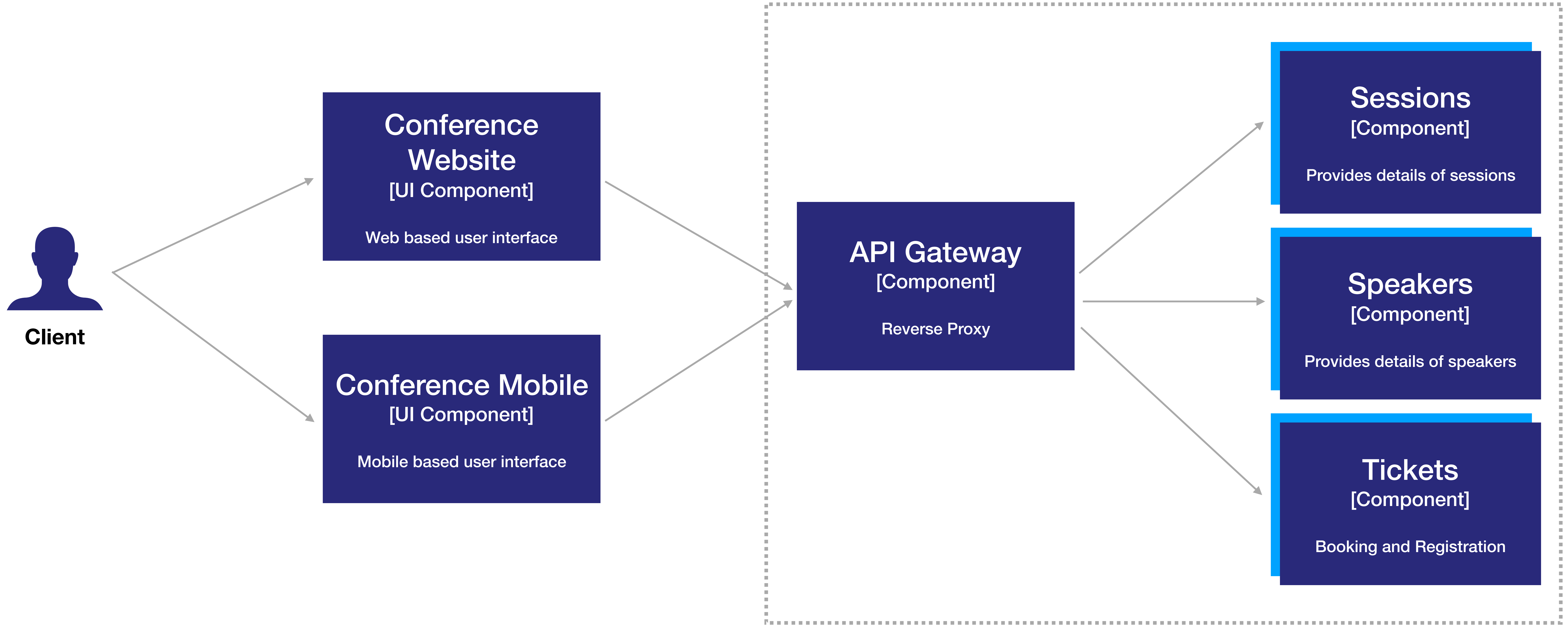
Agenda

- What is a gateway
- Different types of gateway
- Words of caution
- Demo - Applying a gateway to our solution

What is a Gateway?

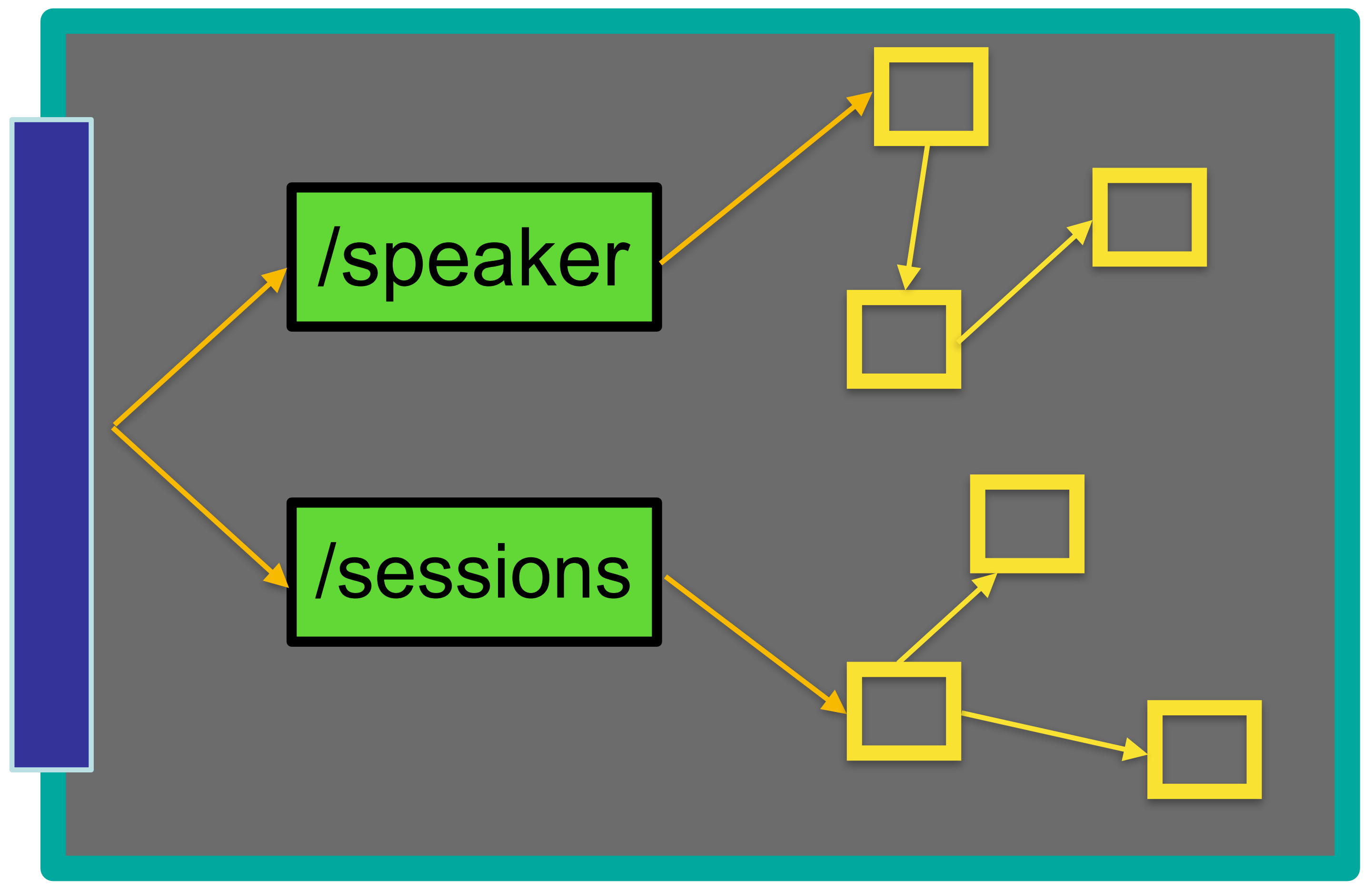


What is a Gateway?



What is a Gateway?

- Provides more functionality than a reverse proxy
- However a gateway it is a reverse proxy too!
- Advanced security mechanisms
- Full control over services discovery and load balancing
- Throttling
- Caching
- Circuit Breaking



Differences Between Gateway

Credit - Ambassador Docs (https://www.getambassador.io/about/microservices-api-gateways/)	Enterprise API Gateway	Microservice Gateway
Goal	Create an API Marketplace	Internal Services
Deployment	Admin API or Team Managed	DevOps Deployed
Metrics	Invocation Rate/HTTP Status	Latency, Traffic
Errors	Custom Errors for Clients	Full Detail of Error
Testing	Staging and Production Promotion	Canary Releases
Development	Docker if Needed	Local Docker/Kubernetes Deployment

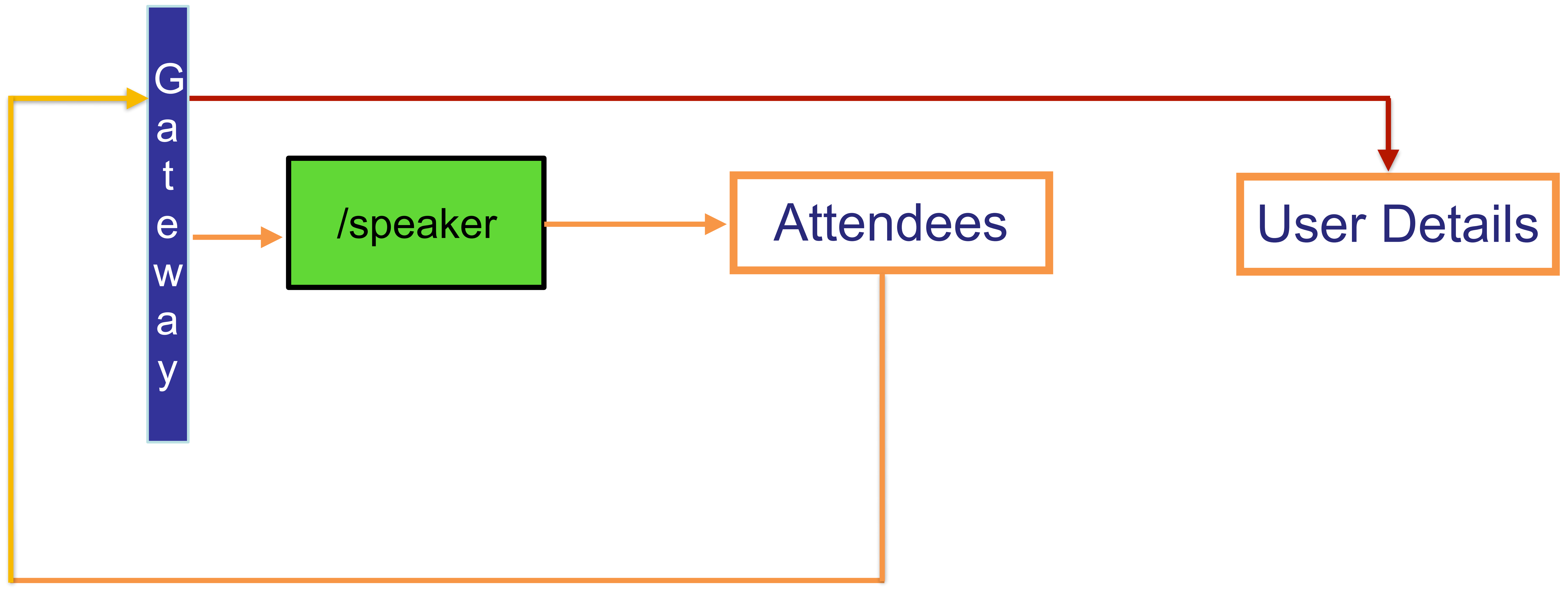
Words of Caution

We remain concerned about business logic and process orchestration implemented in middleware, especially where it requires expert skills and tooling while creating single points of scaling and control. Vendors in the highly competitive API gateway market are continuing this trend by adding features through which they attempt to differentiate their products.

This results in **overambitious API gateway** products whose functionality — on top of what is essentially a reverse proxy — encourages designs that continue to be difficult to test and deploy. API gateways do provide utility in dealing with some specific concerns — such as authentication and rate limiting — but any domain smarts should live in applications or services.

Thoughtworks Technology Radar

Words of Caution



Demo - Applying a Gateway

Creates a Network → task-service

Builds out an image → image

Pulls in a gateway → image

```yaml
version: '3'
services:
  task-service:
    build:
      context: ./
      dockerfile: Dockerfile
    image: "task-service:latest"
    ports:
      - "8080:8080"
  api-gateway:
    image: jpgough/api-workshop-gateway
    ports:
      - "8081:8081"
```

Demo - Applying a Gateway

```
@SpringBootApplication
public class GatewayApplication {

    @Bean
    public RouteLocator customRouteLocator(RouteLocatorBuilder builder) {
        return builder.routes()
                .route("tasks", r -> r.path("/tasks/**")
                .filters(f -> f.rewritePath("/tasks/(?<segment>.*)", "/${segment}"))
                .uri("http://task-service:8080"))
                .build();
    }

    public static void main(String[] args) {
        SpringApplication.run(GatewayApplication.class, args);
    }

}
```