Case Study: Order & Payment Microservices with Zipkin Tracing Business

Scenario :

 An e-commerce company wants to track how requests flow through its microservices to improve debugging and performance monitoring. Currently, when a customer places an order, the request goes through multiple services, and it's hard to identify delays or errors without detailed logging. They decide to implement Spring Cloud Sleuth and Zipkin for distributed tracing

Programs:
payment.java:
```java
package com.example.zipkinpaymentservice.model;


public class Payment {

    private Long orderId;

    private double amount;


    public Payment(Long orderId, double amount) {

        this.orderId = orderId;

        this.amount = amount;

    }


    public Long getOrderId() {

        return orderId;

    }
    public double getAmount() {

        return amount;

    }
}
```

PaymentController:
```java
package com.example.zipkinpaymentservice.controller;
```

```java
import com.example.zipkinpaymentservice.model.Payment;

import com.example.zipkinpaymentservice.service.PaymentService;

import org.springframework.web.bind.annotation.*;


import java.util.List;


@RestController
public class PaymentController {


    private final PaymentService paymentService;


    public PaymentController(PaymentService paymentService) {

        this.paymentService = paymentService;

    }


    @PostMapping("/payments")
    public String processPayment(@RequestBody Payment payment) {

        return paymentService.processPayment(payment);

    }


    @GetMapping("/payments")
    public List<Payment> getPayments() {

        return paymentService.getPayments();

    }
}
```

PaymentService:

```java
package com.example.zipkinpaymentservice.service;
```

```java
import com.example.zipkinpaymentservice.model.Payment;

import io.micrometer.tracing.Tracer;

import io.micrometer.tracing.Span;

import org.springframework.stereotype.Service;


import java.util.ArrayList;

import java.util.List;


@Service
public class PaymentService {

    private final Tracer tracer;
    private final List<Payment> payments = new ArrayList<>(); // in-memory storage


    public PaymentService(Tracer tracer) {
        this.tracer = tracer;
    }


    public String processPayment(Payment payment) {
        payments.add(payment); // store payment


        Span newSpan = tracer.nextSpan().name("process-payment").start();
        try (Tracer.SpanInScope ws = tracer.withSpan(newSpan)) {
            return "Payment processed for Order ID: " + payment.getOrderId();
        } finally {
            newSpan.end();
        }
    }


    public List<Payment> getPayments() {
        return payments;
```

```
    }
}
```

application.properties:

```
spring.application.name=zipkin-payment-service

server.port=8082

management.zipkin.tracing.endpoint=http://localhost:9411/api/v2/spans

management.tracing.sampling.probability=1.0
```

AppConfig:

```java
package com.example.zipkinordersservice.config;


import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.client.RestTemplate;


@Configuration
public class AppConfig {

    @Bean
    public RestTemplate restTemplate() {

        return new RestTemplate();

    }
}
```

Order:
OrderController:

```java
package com.example.zipkinordersservice.controller;

import java.util.List;
```

```java
import    org.springframework.web.bind.annotation.GetMapping;

import    org.springframework.web.bind.annotation.PostMapping;

import    org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;


import com.example.zipkinordersservice.model.Order;

import com.example.zipkinordersservice.service.OrderService;


@RestController
public class OrderController {

    private final OrderService orderService;

    public OrderController(OrderService orderService) {
        this.orderService = orderService;
    }

    @GetMapping("/orders")
    public List<Order> getOrders() {
        return orderService.getOrders();
    }

    @PostMapping("/orders")
    public String placeOrder(@RequestBody Order order) {
        return orderService.placeOrder(order);
    }
}
```

Order.java:
```java
package com.example.zipkinordersservice.model;
```

```java
public class Order {

    private Long id;

    private String productName;

    private double price;


    public Order(Long id, String productName, double price) {

        this.id = id;

        this.productName = productName;

        this.price = price;

    }


    public Long getId() {

        return id;

    }

    public String getProductName() {

        return productName;

    }

    public double getPrice() {

        return price;

    }

}
```

OrderService:

```java
package com.example.zipkinordersservice.service;


import io.micrometer.tracing.Tracer;

import io.micrometer.tracing.Span;


import java.util.ArrayList;
```

```java
import java.util.List;

import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

import com.example.zipkinordersservice.model.Order;

@Service
public class OrderService {

    private final RestTemplate restTemplate;
    private final Tracer tracer;
    private final List<Order> orders = new ArrayList<>();

    public OrderService(RestTemplate restTemplate, Tracer tracer) {
        this.restTemplate = restTemplate;
        this.tracer = tracer;
    }

    public String placeOrder(Order order) {
        orders.add(order); // save order in memory

        Span newSpan = tracer.nextSpan().name("place-order").start();
        try (Tracer.SpanInScope ws = tracer.withSpan(newSpan)) {
            String paymentServiceUrl = "http://localhost:8082/payments";
            String paymentResponse = restTemplate.postForObject(paymentServiceUrl, order, String.class);
            return "Order placed for: " + order.getProductName() + ". " + paymentResponse;
        } finally {
            newSpan.end();
        }
```

```
    }


    public List<Order> getOrders() {

        return orders;

    }
}
```

Application.properties:

spring.application.name=zipkin-orders-service

server.port=8081

management.zipkin.tracing.endpoint=http://localhost:9411/api/v2/spans

management.tracing.sampling.probability=1.0