# ✅ Spring AOP Hands-On Solutions – Customer Model

## 🔹 Hands-On 1: Logging Before Adding Customer

CustomerService.java

@Service  
public class CustomerService {  
 public void addCustomer(Customer customer) {  
 System.out.println("Customer added: " + customer.getName());  
 }  
}

LoggingAspect.java

@Aspect  
@Component  
public class LoggingAspect {  
 @Before("execution(\* com.example.demo.service.CustomerService.addCustomer(..))")  
 public void logBeforeAdd() {  
 System.out.println("Before adding customer - LoggingAspect");  
 }  
}

## 🔹 Hands-On 2: After and AfterReturning Advice for Update Operation

CustomerService.java

public String updateCustomer(Customer customer) {  
 System.out.println("Updating customer: " + customer.getName());  
 return "Update successful";  
}

LoggingAspect.java

@After("execution(\* com.example.demo.service.CustomerService.updateCustomer(..))")  
public void logAfterUpdate() {  
 System.out.println("After updating customer");  
}  
  
@AfterReturning(  
 pointcut = "execution(\* com.example.demo.service.CustomerService.updateCustomer(..))",  
 returning = "result")  
public void logAfterReturningUpdate(String result) {  
 System.out.println("Returned message: " + result);  
}

## 🔹 Hands-On 3: Handling Exceptions Using @AfterThrowing

CustomerService.java

public void deleteCustomer(int id) {  
 if (id <= 0) throw new IllegalArgumentException("Invalid ID");  
 System.out.println("Customer deleted with ID: " + id);  
}

LoggingAspect.java

@AfterThrowing(  
 pointcut = "execution(\* com.example.demo.service.CustomerService.deleteCustomer(..))",  
 throwing = "ex")  
public void logException(Throwable ex) {  
 System.out.println("Exception occurred: " + ex.getMessage());  
}

## 🔹 Hands-On 4: Use @Around to Log Execution Time

CustomerService.java

public void processPurchase() throws InterruptedException {  
 Thread.sleep(1500);  
 System.out.println("Customer purchase processed");  
}

LoggingAspect.java

@Around("execution(\* com.example.demo.service.CustomerService.processPurchase(..))")  
public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {  
 long start = System.currentTimeMillis();  
 Object result = joinPoint.proceed();  
 long end = System.currentTimeMillis();  
 System.out.println("Execution Time: " + (end - start) + " ms");  
 return result;  
}

## 🔹 Hands-On 5: Create a Reusable Pointcut

LoggingAspect.java

@Pointcut("execution(\* com.example.demo.service.CustomerService.\*(..))")  
public void allCustomerMethods() {}  
  
@Before("allCustomerMethods()")  
public void logAll() {  
 System.out.println("A CustomerService method is called");  
}