User Service – JWT Authentication & Profile Management

```java
@RestController
@RequestMapping("/api/users")
@RequiredArgsConstructor
@Slf4j
@SecurityRequirement(name = "bearerAuth")
public class UserController {

    private final UserService userService;

    @PostMapping("/register")
    public ResponseEntity<UserDto> register(@RequestBody RegisterRequest request) {
        log.info("User registration attempt for {}", request.getEmail());
        return ResponseEntity.ok(userService.register(request));
    }

    @PostMapping("/login")
    public ResponseEntity<AuthResponse> login(@RequestBody LoginRequest request) {
        log.info("Login attempt for {}", request.getEmail());
        return ResponseEntity.ok(userService.login(request));
    }

    @GetMapping("/profile")
    public ResponseEntity<UserDto> profile(Authentication auth) {
        return ResponseEntity.ok(userService.getProfile(auth.getName()));
    }

    @PutMapping("/profile")
    public ResponseEntity<UserDto> updateProfile(Authentication auth,
            @RequestBody UpdateProfileRequest request) {
        return ResponseEntity.ok(userService.updateProfile(auth.getName(), request));
    }
}
```

Product Service – Product Catalog & Search

```
@RestController
@RequestMapping("/api/products")
@RequiredArgsConstructor
@Slf4j
@SecurityRequirement(name = "bearerAuth")
public class ProductController {

    private final ProductService productService;

    @GetMapping
    public List<Product> search(@RequestParam(required = false) String query,
                                @RequestParam(required = false) String category) {
        log.info("Search products query={}, category={}", query, category);
        return productService.search(query, category);
    }

    @GetMapping("/{id}")
    public Product getById(@PathVariable Long id) {
        return productService.getById(id);
    }
}
```

Order Service – Order Lifecycle Management

```
@Service
@Slf4j
@RequiredArgsConstructor
public class OrderService {

    private final InventoryClient inventoryClient;
    private final PaymentClient paymentClient;
    private final NotificationClient notificationClient;
    private final OrderRepository orderRepository;

    public Order createOrder(Order order) {

        inventoryClient.validate(order.getItems());

        order.setStatus(OrderStatus.CREATED);
        orderRepository.save(order);

        paymentClient.initiate(order);
        notificationClient.notifyOrderCreated(order);

        log.info("Order {} created", order.getId());
        return order;
    }

    public void handlePaymentCallback(Long orderId, boolean success) {
        Order order = orderRepository.findById(orderId).orElseThrow();

        order.setStatus(success ? OrderStatus.CONFIRMED : OrderStatus.FAILED);
        orderRepository.save(order);

        notificationClient.notifyOrderUpdated(order);
    }
}
```

Payment Service – Secure Payment Processing

```java
@RestController
@RequestMapping("/api/payments")
@RequiredArgsConstructor
@Slf4j
@SecurityRequirement(name = "bearerAuth")
public class PaymentController {

    private final PaymentService paymentService;

    @PostMapping("/initiate")
    public PaymentResponse initiate(@RequestBody PaymentRequest request) {
        log.info("Payment initiation for order {}", request.getOrderId());
        return paymentService.process(request);
    }

    @PostMapping("/callback")
    public void callback(@RequestBody PaymentCallback callback) {
        log.info("Payment callback received");
        paymentService.handleCallback(callback);
    }
}
```

Inventory Service – Stock Validation & Updates

```
@RestController
@RequestMapping("/api/inventory")
@RequiredArgsConstructor
@Slf4j
@SecurityRequirement(name = "bearerAuth")
public class InventoryController {

    private final InventoryService inventoryService;

    @PostMapping("/validate")
    public void validate(@RequestBody List<OrderItem> items) {
        inventoryService.validateStock(items);
    }

    @PostMapping("/release")
    public void release(@RequestBody List<OrderItem> items) {
        inventoryService.releaseStock(items);
    }
}
```

Notification Service – Email & SMS Notifications

```
@RestController
@RequestMapping("/api/notifications")
@RequiredArgsConstructor
@Slf4j
@SecurityRequirement(name = "bearerAuth")
public class NotificationController {

    private final NotificationService notificationService;

    @PostMapping("/order")
    public void notify(@RequestBody NotificationRequest request) {
        notificationService.send(request);
        log.info("Notification sent for order {}", request.getOrderId());
    }
}
```

API Gateway – JWT Enforcement

```java
@EnableWebFluxSecurity
public class SecurityConfig {

    @Bean
    public SecurityWebFilterChain security(ServerHttpSecurity http) {
        return http
            .authorizeExchange(ex -> ex.anyExchange().authenticated())
            .oauth2ResourceServer(ServerHttpSecurity.OAuth2ResourceServerSpec::jwt)
            .build();
    }
}
```