**3. Microservices composite handson Solutions**

**1. Account Microservice**

@RestController

@RequestMapping("/accounts")

public class AccountController {

@GetMapping("/{number}")

public Map<String, Object> getAccount(@PathVariable String number) {

Map<String, Object> response = new HashMap<>();

response.put("number", number);

response.put("type", "savings");

response.put("balance", 234343);

return response;

}

}

**2. Loan Microservice**

{ "number": "H00987987972342", "type": "car", "loan": 400000, "emi": 3258, "tenure": 18 }

@RestController

@RequestMapping("/loans")

public class LoanController {

@GetMapping("/{number}")

public Map<String, Object> getLoan(@PathVariable String number) {

Map<String, Object> response = new HashMap<>();

response.put("number", number);

response.put("type", "car");

response.put("loan", 400000);

response.put("emi", 3258);

response.put("tenure", 18);

return response;

}

}

**3. Eureka Discovery Server**

@SpringBootApplication

@EnableEurekaServer

public class EurekaDiscoveryServerApplication {

public static void main(String[] args) {

SpringApplication.run(EurekaDiscoveryServerApplication.class, args);

}

}

**4. Register Account and Loan Services with Eureka**

@SpringBootApplication

@EnableDiscoveryClient

public class AccountApplication { ... }

@SpringBootApplication

@EnableDiscoveryClient

public class LoanApplication { ... }

**5. Greet-Service Microservice**

@RestController

public class GreetController {

@GetMapping("/greet")

public String greet() {

return "Hello World";

}

}

@SpringBootApplication

@EnableDiscoveryClient

public class GreetServiceApplication {

public static void main(String[] args) {

SpringApplication.run(GreetServiceApplication.class, args);

}

}

**6. API Gateway with Eureka Integration & Global Filter Logging**

@Component

public class LogFilter implements GlobalFilter, Ordered {

private static final Logger logger = LoggerFactory.getLogger(LogFilter.class);

@Override

public Mono<Void> filter(ServerWebExchange exchange, GatewayFilterChain chain) {

logger.info("Incoming request: " + exchange.getRequest().getURI());

return chain.filter(exchange);

}

@Override

public int getOrder() {

return -1; // High priority

}

}

@SpringBootApplication

@EnableDiscoveryClient

public class ApiGatewayApplication {

public static void main(String[] args) {

SpringApplication.run(ApiGatewayApplication.class, args);

}

}