Week 5 ASSIGNMENT – MICROSERVICES

**AccountController.java**

@RestController

@RequestMapping("/accounts")

public class AccountController {

@GetMapping("/{number}")

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

return Map.of(

"number", number,

"type", "savings",

"balance", 234343

);

}

}

**LoanController.java**

@RestController

@RequestMapping("/loans")

public class LoanController {

@GetMapping("/{number}")

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

return Map.of(

"number", number,

"type", "car",

"loan", 400000,

"emi", 3258,

"tenure", 18

);

}

}

**EurekaServer:**

@SpringBootApplication

@EnableEurekaServer

public class EurekaDiscoveryServerApplication {

public static void main(String[] args) {

SpringApplication.run(EurekaDiscoveryServerApplication.class, args);

}

}

**application.properties:**

server.port=8761

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

logging.level.com.netflix.eureka=OFF

logging.level.com.netflix.discovery=OFF

**GreetController.java**

@RestController

public class GreetController {

@GetMapping("/greet")

public String greet() {

return "Hello World";

}

}

**LogFilter.java(API Gateway)**

@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("Request: " + exchange.getRequest().getMethod() + " " + exchange.getRequest().getURI());

return chain.filter(exchange);

}

@Override

public int getOrder() {

return -1;

}

}