**Creating Microservices for Account and Loan**

**1. Account Microservice**

Create folder: D:\<employee\_id>\microservices

Open [https://start.spring.io/](https://start.spring.io/" \t "_new)

Group: com.cognizant, Artifact: account

Dependencies: Spring Boot DevTools, Spring Web

Download, extract, and place in microservices folder

In command prompt, run: mvn clean package

Import in Eclipse

Create controller:

GET /accounts/{number}

Response:

json

CopyEdit

{ "number": "00987987973432", "type": "savings", "balance": 234343 }

Run and test

**2. Loan Microservice**

Same steps as Account

Artifact: loan

Controller:

GET /loans/{number}

Response:

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

Change port to avoid conflict:

server.port=8081 in application.properties

**Eureka Discovery Server Setup**

**1. Create Eureka Server**

Go to [https://start.spring.io/](https://start.spring.io/" \t "_new)

Group: com.cognizant, Artifact: eureka-discovery-server

Dependency: Eureka Server

Download and build with Maven

Import into Eclipse

Annotate main class with @EnableEurekaServer

Add to application.properties:

ini

CopyEdit

server.port=8761eureka.client.register-with-eureka=falseeureka.client.fetch-registry=falselogging.level.com.netflix.eureka=OFFlogging.level.com.netflix.discovery=OFF

Run and open: [http://localhost:8761](http://localhost:8761" \t "_new)

**2. Register Account Service**

Modify existing account service:

Add dependencies: Eureka Discovery Client

Annotate with @EnableDiscoveryClient

Add:

spring.application.name=account-service

Restart Eureka, then start account service

Refresh: [http://localhost:8761](http://localhost:8761" \t "_new) — should list account-service

**3. Register Loan Service**

Repeat above steps for loan-service

**Creating API Gateway with Logging Filter**

**1. Greet Microservice**

Create via [https://start.spring.io/](https://start.spring.io/" \t "_new)

Dependencies: Spring Web

Controller:

java

CopyEdit

@RestControllerpublic class GreetController {

@GetMapping("/greet")

public String greet() {

return "Hello World";

}

}

application.properties:

ini

CopyEdit

spring.application.name=greet-serviceserver.port=8082

Run and test: http://localhost:8082/greet

**2. Eureka Server**

Same as previous Eureka setup (port 8761)

**3. Register Greet Service**

Add Eureka Client dependency to greet-service

Copy Spring Cloud version and dependency-management section from Eureka Server pom.xml

Annotate main class with @EnableDiscoveryClient

Restart Eureka, then restart greet-service

Verify at [http://localhost:8761](http://localhost:8761" \t "_new) — should show greet-service

**4. Create API Gateway**

Via Spring Initializer

Artifact: api-gateway

Dependencies: Spring Cloud Gateway, Eureka Discovery Client

application.properties:

ini

CopyEdit

spring.application.name=api-gatewayserver.port=9090eureka.client.service-url.defaultZone=http://localhost:8761/eureka/spring.cloud.gateway.discovery.locator.enabled=truespring.cloud.gateway.discovery.locator.lowerCaseServiceId=true

**5. Access Greet Through Gateway**

URL: http://localhost:9090/greet-service/greet

**6. Create Global Logging Filter**

In api-gateway project, create LogFilter.java:

@Componentpublic class LogFilter implements GlobalFilter {

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

@Override

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

logger.info("Request path: " + exchange.getRequest().getPath());

return chain.filter(exchange);

}

}

Run and access: http://localhost:9090/greet-service/greet

Check console logs for request path