

Elaboro	Documento	Versión	Descripción	Fecha
Luis Sánchez Martínez	Práctica integración FeignClient.	1.0	Creación de documento	10/04/2024
Luis Sánchez Martínez	Práctica integración FeignClient.	1.1	Se agrega ErrorDecoder y CircuitBreaker en el FeignClient.	13/04/2024

## Práctica integración FeignClient

Tiempo estimado: 3 horas.

Uso de:

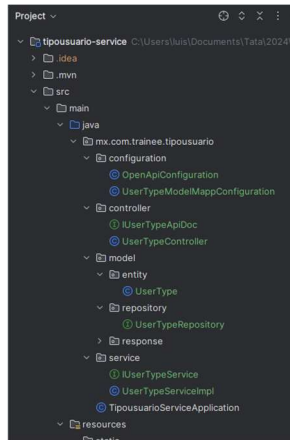
- JDK 17
- Maven
- IntelliJ
- Postman
- Git
- Spring Cloud
  - Resilience4J
  - Actuator
  - FeignClient

Crear Api y Microservicio realizara el CRUD a la tabla TipoUsuario.

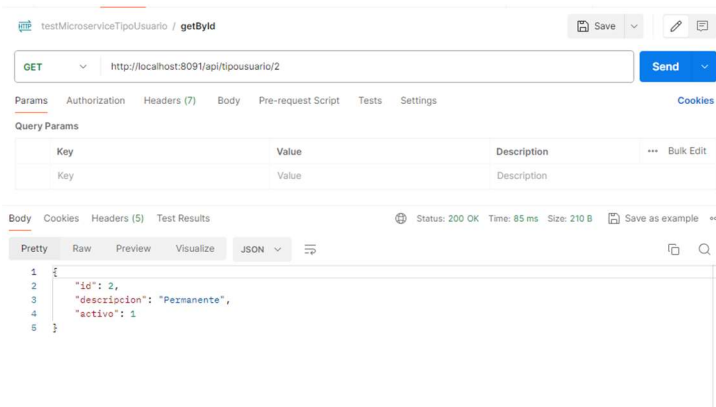
1. Se crea el API para el CRUD de *tipousuario-service*.



2. Se crea el proyecto en *Spring Boot* donde se implementará el microservicio *tipousuario-service*.



3. Levantar el servicio *tipousuario-service* y ejecutar la petición del endpoint GET *getById*.



Nota: bajar el proyecto de la ruta en git <https://github.com/luis-s-mtz/tipousuario-service.git>, bajar de la rama *development*.

Integrar FeignClien en el proyecto de Spring Boot permiso-service.

1. Agregar la dependencia *spring-cloud-starter-openfeign* en el pom del proyecto *permiso-service*.

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-openfeign</artifactId>
</dependency>
```

```
77
78
79
80
81
82
83
84
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-openfeign</artifactId>
</dependency>
</dependencies>
<dependencyManagement>
  <dependencies>
```

2. Colocar la notación *@EnableFeignClients* en la clase *PermisoServiceApplication*.

```

@EnableFeignClients
@SpringBootApplication
public class PermisoServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(PermisoServiceApplication.class, args);
    }

}

```

```

@EnableFeignClients
@SpringBootApplication
public class PermisoServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(PermisoServiceApplication.class, args);
    }

}

```

3. Definir una interface, agregarle la notación `@FeignClient` y definir el método para consumir el servicio que mandara a llamar *tipousuario-service*.

```

@FeignClient(name = "tipousuario-service", path = "/api/tipousuario"
, url = "localhost:8091")
public interface UserTypeFeignClient {

    @GetMapping("/{id}")
    ResponseEntity<UserTypeDTO> getById(@PathVariable(value = "id") Integer id);
}

```

```

package mx.com.tcs.permiso.client;

import java.util.List;

/**
 * @author luis
 * @since 1.0
 * Interface to create feign client to request tipousuario-service and get the description of the UserType object
 */
3 usages new *
@FeignClient(name = "tipousuario-service",
url = "localhost:8091")
@RequestMapping("/api/tipousuario")
public interface UserTypeFeignClient {

    1 usage new *
    @GetMapping("/{id}")
    ResponseEntity<UserTypeDTO> getById(@PathVariable(value = "id") Integer id);
}

```

4. Modificar los scripts de creación de esquema e inserción de registros para integrar una tabla relacional que integre los datos del identificador del permiso relacionado con el identificador del tipo de usuario.

```

DROP TABLE IF EXISTS Permiso;
CREATE TABLE Permiso (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(25) NOT NULL,
  descripcion VARCHAR(50) NOT NULL,
  activo INT NOT NULL,
  id_Padre INT NOT NULL,
  icono VARCHAR(50) NOT NULL
);
DROP TABLE IF EXISTS Tipo_Usuario_Permiso;
CREATE TABLE Tipo_Usuario_Permiso (
  id INT AUTO_INCREMENT PRIMARY KEY,
  id_Tipo_Usuario INT NOT NULL,
  id_Permiso INT NOT NULL,
  activo INT NOT NULL
);

```

```

INSERT INTO Permiso VALUES (1,'Academico', 'Categoria de permisos
academicos', 1, 0, '/images/icon_school.gif');
INSERT INTO Permiso VALUES (2,'Enfermedad', 'Categoria de permisos por
enfermedad', 1, 0, '/images/icon_disease.gif');
INSERT INTO Permiso VALUES (3,'Permiso academ por examen', 'Presentacion de
exámenes', 1, 1, '/images/icon_school_quiz.gif');
INSERT INTO Permiso VALUES (4,'Permiso por enfermedad', 'Consulta en
Clinica', 1, 2, '/images/icon_check_health.gif');
INSERT INTO Tipo_Usuario_Permiso VALUES (1,1,4,1);
INSERT INTO Tipo_Usuario_Permiso VALUES (2,2,3,1);
INSERT INTO Tipo_Usuario_Permiso VALUES (3,2,4,1);

```

5. Agregar la lógica en la capa de servicio para mandar a llamar el cliente que se utilizara para el servicio tipousuario-service.

```

/**
 * Method to return all records of Permiso object find by Id TipoUsuario.
 * @param idTipoUsuario The identifier of TipoUsuario entity object.
 * @return A PermisoTipoUsuarioDTO object.
 */
ResponseEntity<PermisoTipoUsuarioDTO> findByParams(Integer idTipoUsuario);

```

```

6 usages 1 implementation 1 Trainer
15 public interface IPermisoService {
16
17     /**
18      * Method to return all records of Permiso object.
19      * @return a response entity of list or Permiso object.
20      */
21     5 usages 1 implementation 1 Trainer
22     ResponseEntity<List<PermisoDTO>> listAll();
23
24     /**
25      * Method to return all records of Permiso object find by Id TipoUsuario.
26      * @param idTipoUsuario The identifier of TipoUsuario entity object.
27      * @return A PermisoTipoUsuarioDTO object.
28      */
29     1 usage 1 implementation new *
30     ResponseEntity<PermisoTipoUsuarioDTO> findByParams(Integer idTipoUsuario);
31 }

```

```

@Override
public ResponseEntity<PermisoTipoUsuarioDTO> findByParams(Integer idTipoUsuario) {

    PermisoTipoUsuarioDTO permisoTipoUsuarioDTO;
    ResponseEntity<UserTypeDTO> respUserTypeDTO = userTypeFeignClient.getById(idTipoUsuario);

    String tipoUsuario = respUserTypeDTO.getStatusCode().equals(HttpStatus.OK) ?
        respUserTypeDTO.getBody().getDescription() :
        null;

    if (tipoUsuario == null) {
        log.error("Error in [{}] findByParams(): {}",
            this.getClass().getName(), "Error al consultar informacion de feign client.");
        throw new PermisoSrvInternalServerErrorException("Error al consultar informacion de
feign client.");
    } else {
        permisoTipoUsuarioDTO = new PermisoTipoUsuarioDTO();
        permisoTipoUsuarioDTO.setTipoUsuario(tipoUsuario);
        List<TipoUsuarioPermiso> relationList = findPermisoByIdTipoUsuario(idTipoUsuario);

        List<Integer> ids = new ArrayList<>();
        relationList.stream().
            map(tipoUsuarioPermiso -> ids.add(tipoUsuarioPermiso.getIdPermiso())).
            toList();

        permisoTipoUsuarioDTO.setPermisos(getPermisosByIds(ids));
    }
    return ResponseEntity.ok(permisoTipoUsuarioDTO);
}

```

6. Agregar el endpoint GET *findByParams* en la clase *PermisoController* que permitirá obtener la lista de permisos relacionados con el tipo de usuario.

```

37  /**
38   * Endpoint used to get all records in Permiso entity.
39   * @return a ResponseEntity of the list of Permiso object.
40   */
41   @Override
42   public ResponseEntity<List<PermisoDTO>> getAll() { return permisoService.listAll(); }
43
44   /**
45   * Endpoint used to get all records in Permiso entity find by idTipoUsuario.
46   *
47   * @param idTipoUsuario The identifier of the Usuario entity.
48   * @return A ResponseEntity of the PermisoTipoUsuarioDTO object.
49   */
50   @Override
51   public ResponseEntity<PermisoTipoUsuarioDTO> findByParams(Integer idTipoUsuario) {
52       return permisoService.findByParams(idTipoUsuario);
53   }
54 }

```

7. Agregar la configuración al Slf4J en el archivo de propiedades.

```

logging.level.root=warn
logging.level.org.springframework.web=debug
logging.level.org.hibernate=error

```

```

25 management.endpoint.metrics.enabled=true
26 management.endpoints.web.exposure.include=metrics,info,health
27 management.endpoint.health.show-components=always
28 management.endpoint.health.show-details=always
29
30 spring.cloud.circuitbreaker.resilience4j.enabled=true
31
32 logging.level.root=warn
33 logging.level.org.springframework.web=debug
34 logging.level.org.hibernate=error

```

8. Levantar el servicio de *permiso-service* y realizar la prueba ejecutando el endpoint de consulta de lista de la tabla permiso usando el id de TipoUsuario.

testMicroservicePermisos / getPermisoByParams

GET http://localhost:8090/api/permiso/findByParams/2

Status: 200 OK Time: 47 ms Size: 496 B

```

1 {
2   "tipoUsuario": "Permanente",
3   "permisos": [
4     {
5       "id": 3,
6       "nombre": "Permiso academ por examen",
7       "descripcion": "Presentacion de examenes",
8       "activo": 1,
9       "idPadre": 1,
10      "icono": "/images/icon_school_quiz.gif"
11    },
12    {
13      "id": 4,
14      "nombre": "Permiso por enfermedad",
15      "descripcion": "Consulta en Clinica",

```

9. Revisar los logs para identificar la llamada al servicio tipousuario-service utilizando FeignClient.

```

PermisoServiceApplication
:
RequestMappingHandlerMapping : Mapped to mx.com.tcs.permiso.controller.PermisoController.findById(Integer)
HandlerInterceptor : Opening JPA EntityManager in OpenEntityManagerInViewInterceptor
HandlerInterceptor : Explicit slash decoding specified, decoding all slashes in url
HttpURLConnection : sun.net.www.MessageHeader@44a58a2a5 pairs: {GET /api/tipousuario/2 HTTP/1.1: null}{Accept: */*}{User-Agent: Java/17.0.10}{Host: 1
HttpURLConnection : sun.net.www.MessageHeader@3880e15c6 pairs: {null: HTTP/1.1 200}{Content-Type: application/json}{Transfer-Encoding: chunked}{Date:
HttpEntityConverter : Reading to [mx.com.tcs.permiso.model.client.UserTypeDTO]
EntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
HandlerInterceptor : Writing [PermisoTipousuarioDTO(tipousuario=Permanente, permisos=[PermisoDTO(id=3, nombre=Permiso academico e (truncated)...)]
HandlerInterceptor : Closing JPA EntityManager in OpenEntityManagerInViewInterceptor
DispatcherServlet : Completed 200 OK
pool.HikariPool : HikariPool-1 - Pool stats (total=10, active=0, idle=10, waiting=0)
pool.HikariPool : HikariPool-1 - Fill pool skipped, pool has sufficient level or currently being filled (queueDepth=0).

exec-2] m.s.m.a.ResponseBodyMethodProcessor : Writing [(123, 34, 111, 112, 101, 110, 97, 112, 105, 34, 58, 34, 51, 40, 48, 40, 49, 34, 44, 34, 105, 1
exec-2] o.s.web.servlet.DispatcherServlet : Completed 200 OK
exec-4] o.s.web.servlet.DispatcherServlet : GET "/api/tipousuario/2", parameters={}
exec-4] m.s.m.a.RequestMappingHandlerMapping : Mapped to mx.com.trainee.tipousuario.controller.UserTypeController.getById(Integer)
exec-4] o.s.w.s.m.a.HttpEntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
exec-4] o.s.w.s.m.a.HttpEntityMethodProcessor : Writing [UserTypeDTO(id=2, description=Permanente, status=1)]
exec-4] o.s.web.servlet.DispatcherServlet : Completed 200 OK
exec-6] o.s.web.servlet.DispatcherServlet : GET "/api/tipousuario/2", parameters={}
exec-6] m.s.m.a.RequestMappingHandlerMapping : Mapped to mx.com.trainee.tipousuario.controller.UserTypeController.getById(Integer)
exec-6] o.s.w.s.m.a.HttpEntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
exec-6] o.s.w.s.m.a.HttpEntityMethodProcessor : Writing [UserTypeDTO(id=2, description=Permanente, status=1)]
exec-6] o.s.web.servlet.DispatcherServlet : Completed 200 OK

```

10. Integrar la clase *CustomErrorDecoder* y *FeignConfiguration* para recuperar las respuestas desde la solicitud desde *FeignClient* y manejar las excepciones lanzadas 404 y 500 desde el servicio *tipousuario-service*.

```

public class CustomErrorDecoder implements ErrorDecoder {
    /**
     * Method to manage the errors generated in FeignClient.
     *
     * @param s The methodKey.
     * @param response The response of the FeignClient.
     * @return The exception throw.
     */
    @Override
    public Exception decode(String s, Response response) {
        ErrorMessageDTO feignClientErrorDTO = deserializeResponse(response);
        String errorMessage = feignClientErrorDTO.getError();

        switch (response.status()) {
            case 400:
                return new BadRequestException();
            case 404:
                return new ItemNotFoundException("Not found in request to FeignClient: " + errorMessage + ".");
            case 500:
                return new PermisoSrvInternalServErrorException(
                    "Response from FeignClient return an Internal Server Error: " + errorMessage + ".");
            default:
                return new PermisoSrvInternalServErrorException("Exception in request from FeignClient: " +
                    errorMessage + ".");
        }
    }

    private ErrorMessageDTO deserializeResponse(Response response) {
        ErrorMessageDTO errorDTO = null;
        try (InputStream bodyIs = response.body().asInputStream()) {
            ObjectMapper mapper = new ObjectMapper();
            errorDTO = mapper.readValue(bodyIs, ErrorMessageDTO.class);
        } catch (IOException e) {
            throw new PermisoSrvInternalServErrorException("Error when deserialize response from FeignClient: " +
                e.getMessage());
        }
        return errorDTO;
    }
}

```

```

@Slf4j
public class CustomErrorDecoder implements ErrorDecoder {
    /**
     * Method to manage the errors generated in FeignClient.
     *
     * @param s The methodKey.
     * @param response The response of the FeignClient.
     * @return The exception throw.
     */
    // Trainer
    @Override
    public Exception decode(String s, Response response) {
        ErrorMessageDTO feignClientErrorDTO = deserializeResponse(response);
        String errorMessage = feignClientErrorDTO.getError();

        switch (response.status()) {
            case 400:
                return new BadRequestException();
            case 404:
                return new ItemNotFoundException("Not found in request to FeignClient: " + errorMessage + ".");
            case 500:
                return new PersoSrvInternalServerErrorException(
                    "Response from FeignClient return an Internal Server Error: " + errorMessage + ".");
            default:
                return new PersoSrvInternalServerErrorException("Exception in request from FeignClient: " + errorMessage + ".");
        }
    }
}

```

```

public class FeignConfiguration {

    /**
     * Method to create the Bean of ErrorDecoder.
     * @return An ErrorDecoder Bean.
     */
    @Bean
    ErrorDecoder errorDecoder() {
        return new CustomErrorDecoder();
    }
}

```

```

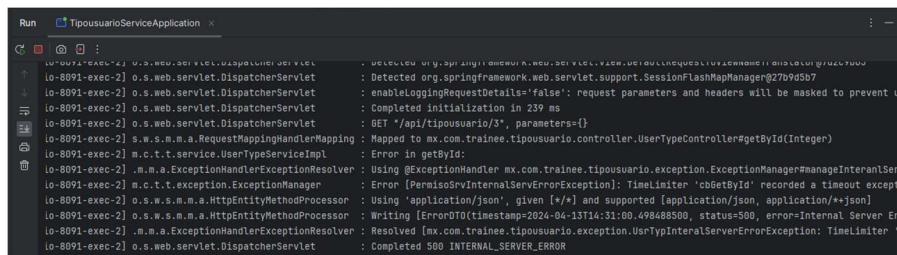
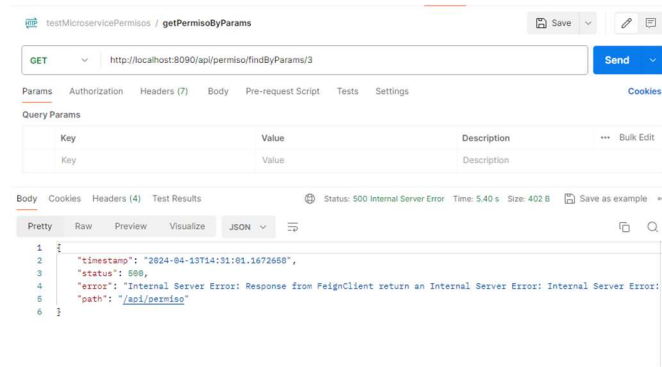
/**
 * @author luis
 * @since 1.0
 * Configuration class to create an ErrorDecoder Bean using the CustomerErrorDecoder.
 */
// 2 usages 1 Trainer
@Configuration
public class FeignConfiguration {

    /**
     * Method to create the Bean of ErrorDecoder.
     * @return An ErrorDecoder Bean.
     */
    // no usages 1 Trainer
    @Bean
    ErrorDecoder errorDecoder() { return new CustomErrorDecoder(); }
}

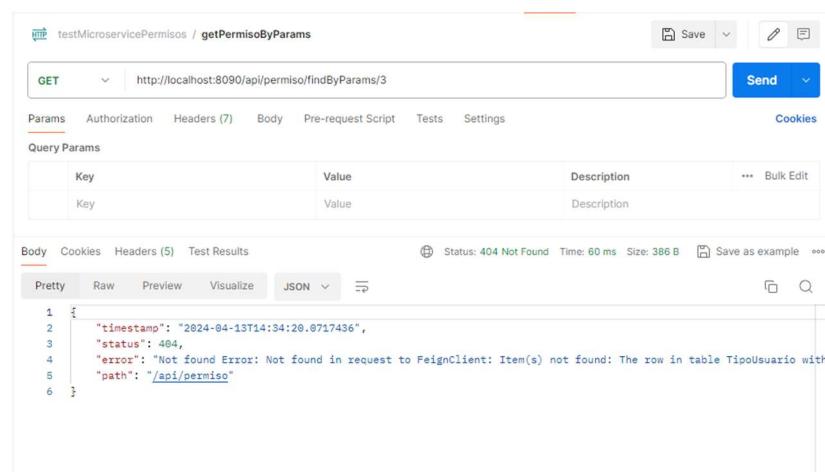
```

11. Ejecutar la validación del error 500 lanzado desde el cliente tipousuario-service.





12. Realizar pruebas para validar el manejo de ErrorDecoder con un error 404 recuperado desde el FeignClient.



```
Run PermisServiceApplication
o.s.web.servlet.DispatcherServlet : GET "/api/permiso/findByParams/3", parameters={}
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to mx.com.tcs.permiso.controller.PermisoController#findByParams(Integer)
.m.m.a.ExceptionHandlerExceptionHandlerResolver : Using @ExceptionHandler mx.com.tcs.permiso.exception.ExceptionManager#manageNotFoundException(ItemNotFoundException(ItemNotFoun
m.c.t.p.exception.ExceptionManager : Error [ItemNotFoundException]: Not found in request to FeignClient: Item(s) not found: The row in table TipoUs
o.s.w.s.m.m.a.HttpEntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
.m.m.a.ExceptionHandlerExceptionHandlerResolver : Writing [ErrorDTO(timestamp=2024-04-13T14:34:20.071743600, status=404, error=Not found Error: Not found in re
o.s.web.servlet.DispatcherServlet : Resolved [mx.com.tcs.permiso.exception.ItemNotFoundException: Not found in request to FeignClient: Item(s) not
o.s.web.servlet.DispatcherServlet : Completed 404 NOT_FOUND
```

```
Run TipousuarioServiceApplication
io-8091-exec-1 o.s.web.servlet.DispatcherServlet : GET "/api/tipousuario/3", parameters={}
io-8091-exec-1 s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to mx.com.trainee.tipousuario.controller.UserTypeController#getId(Integer)
io-8091-exec-1 .m.m.a.ExceptionHandlerExceptionHandlerResolver : Using @ExceptionHandler mx.com.trainee.tipousuario.exception.ExceptionManager#manageNotFoundException
m.c.t.p.exception.ExceptionManager : Error [ItemNotFoundException]: The row in table TipoUsuario with id 3 is not found.
io-8091-exec-1 o.s.w.s.m.m.a.HttpEntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
io-8091-exec-1 .m.m.a.ExceptionHandlerExceptionHandlerResolver : Writing [ErrorDTO(timestamp=2024-04-13T14:34:20.057230200, status=404, error=Item(s) not found:
io-8091-exec-1 .m.m.a.ExceptionHandlerExceptionHandlerResolver : Resolved [mx.com.trainee.tipousuario.exception.ItemNotFoundException: The row in table TipoUsuar
io-8091-exec-1 o.s.web.servlet.DispatcherServlet : Completed 404 NOT_FOUND
```

13. Integrar la configuración en el archivo de propiedades para el uso de CircuitBreaker en el cliente de UserTypeFeignClient.

```
spring.cloud.openfeign.circuitbreaker.enabled=true
spring.cloud.openfeign.circuitbreaker.alphanumeric-ids.enabled=true
resilience4j.circuitbreaker.instances.UserTypeFeignClient.getId.minimumNumberOfCalls=50
resilience4j.timelimiter.instances.UserTypeFeignClient.getId.timeoutDuration=2s
```

```
32 logging.level.root=warn
33 logging.level.org.springframework.web=debug
34 logging.level.org.hibernate=error
35
36 spring.cloud.openfeign.circuitbreaker.enabled=true
37 spring.cloud.openfeign.circuitbreaker.alphanumeric-ids.enabled=true
38 resilience4j.circuitbreaker.instances.UserTypeFeignClient.getId.minimumNumberOfCalls=50
39 resilience4j.timelimiter.instances.UserTypeFeignClient.getId.timeoutDuration=2s
```

14. Agregar la clase UserTypeFallback para el manejo de la petición en caso de que el circuito sea abierto y no se obtenga respuesta del cliente.

```
@Component
public class UserTypeFallback implements UserTypeFeignClient{
    /**
     * Method used to get UserType record find by Id.
     * @param id Identifier of UserType objet to find.
     * @return An ResponseEntity object of UserTypeDTO.
     */
    @Override
    public ResponseEntity<UserTypeDTO> getId(Integer id) {
        throw new PermisoSrvInternalServerErrorException("When request to UserTypeFeignClient.");
    }
}
```

```

8  /**
9   * @author Luis
10  * @since 1.0
11  *
12  * Component class to implements fallback when UserTypeFeignClient use the CircuitBreaker
13  * pattern.
14  */
15  @Component
16  public class UserTypeFallback implements UserTypeFeignClient {
17      /**
18       * Method used to get UserType record find by Id.
19       * @param id Identifier of UserType object to find.
20       * @return An ResponseEntity object of UserType0.
21       */
22      @Override
23      public ResponseEntity<UserType0> getById(Integer id) {
24          throw new PermisoSrvInternalServerErrorException("When request to UserTypeFeignClient.");
25      }
26  }

```

15. Levantar el servicio permiso-service para validar la implementación de CircuitBreaker en el cliente de UserTypeFeignClient.

```

Run PermisServiceApplication
C:\Program Files\Java\jdk-17\bin\java.exe
...
:: Spring Boot :: (v3.2.4)

2024-04-13T22:46:35.084-06:00 DEBUG --- [permiso-service] [main] o.s.w.f.ServerHttpObservationFilter : Filter 'webMvcObservationFilter'
2024-04-13T22:46:38.686-06:00 WARN 15480 --- [permiso-service] [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. This can lead to stale data being read without a reload trigger.
2024-04-13T22:46:39.276-06:00 DEBUG 15480 --- [permiso-service] [main] s.w.s.m.m.a.RequestMappingHandlerMapping : 8 mappings in 'requestMappingHandlerMapping'
2024-04-13T22:46:39.578-06:00 DEBUG 15480 --- [permiso-service] [main] o.s.w.s.handler.SimpleUrlHandlerMapping : Patterns [/webjars/**, /**, /]
2024-04-13T22:46:39.697-06:00 DEBUG 15480 --- [permiso-service] [main] s.w.s.m.m.a.RequestMappingHandlerAdapter : ControllerAdvice beans: 0 @ModelAttribute, 0 @ExceptionHandler
2024-04-13T22:46:39.803-06:00 DEBUG 15480 --- [permiso-service] [main] .m.m.a.ExceptionHandlerExceptionResolver : ControllerAdvice beans: 2 @ExceptionHandler

```

16. Ejecutar peticiones a la función GET /api/permiso/findByParam/{id} con el servicio tipousuario-service apagado para validar el funcionamiento del circuito abierto y la ejecución de la clase Fallback.

```

testMicroservicePermisos / getPermisoByParams
GET http://localhost:8090/api/permiso/findByParams/3
Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies
Query Params
Key Value Description Bulk Edit
Key Value Description

Body Cookies Headers (4) Test Results Status: 500 Internal Server Error Time: 19 ms Size: 302 B Save as example
Pretty Raw Preview Visualize JSON
1 {
2   "timestamp": "2024-04-13T22:46:54.4536362",
3   "status": 500,
4   "errors": "Internal Server Error: When request to UserTypeFeignClient.",
5   "path": "/api/permiso"
6 }

```

```

Run PermisServiceApplication
-10] o.s.web.servlet.DispatcherServlet : GET "/api/permiso/findByParams/3", parameters={}
-10] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to mx.com.tcs.permiso.controller.PermisoController#findByParams(Integer)
-10] .m.m.a.ExceptionHandlerExceptionResolver : Using @ExceptionHandler mx.com.tcs.permiso.exception.ExceptionManager#manageInternalServerErrorExcept(Permis
-10] m.c.t.p.exception.ExceptionManager : Error [PermisoSrvInternalServerErrorException]: When request to UserTypeFeignClient.
-10] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
-10] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Writing [ErrorResponse(timestamp=2024-04-13T22:46:54.4536362, status=500, error=Internal Server Error: When r
-10] .m.m.a.ExceptionHandlerExceptionResolver : Resolved [mx.com.tcs.permiso.exception.PermisoSrvInternalServerErrorException: When request to UserTypeFeign
-10] o.s.web.servlet.DispatcherServlet : Completed 500 INTERNAL_SERVER_ERROR
c-1] o.s.web.servlet.DispatcherServlet : GET "/api/permiso/findByParams/3", parameters={}
c-1] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to mx.com.tcs.permiso.controller.PermisoController#findByParams(Integer)
c-1] .m.m.a.ExceptionHandlerExceptionResolver : Using @ExceptionHandler mx.com.tcs.permiso.exception.ExceptionManager#manageInternalServerErrorExcept(Permis
c-1] m.c.t.p.exception.ExceptionManager : Error [PermisoSrvInternalServerErrorException]: When request to UserTypeFeignClient.
c-1] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Using 'application/json', given [*/] and supported [application/json, application/*+json]
c-1] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Writing [ErrorResponse(timestamp=2024-04-13T22:46:54.4536362, status=500, error=Internal Server Error: When r
c-1] .m.m.a.ExceptionHandlerExceptionResolver : Resolved [mx.com.tcs.permiso.exception.PermisoSrvInternalServerErrorException: When request to UserTypeFeign
c-1] o.s.web.servlet.DispatcherServlet : Completed 500 INTERNAL_SERVER_ERROR

```

## Referencias

<https://docs.spring.io/spring-cloud-openfeign/docs/current/reference/html/>

<https://barcochrist.medium.com/comunicaci%C3%B3n-entre-dos-micro-servicios-con-feign-y-eureka-ab4f3daf70c6>

[https://cloud.spring.io/spring-cloud-netflix/multi/multi\\_spring-cloud-feign.html](https://cloud.spring.io/spring-cloud-netflix/multi/multi_spring-cloud-feign.html)

<https://docs.spring.io/spring-boot/docs/2.1.13.RELEASE/reference/html/boot-features-logging.html>

<https://www.baeldung.com/spring-boot-logging>

<https://www.baeldung.com/java-feign-client-exception-handling>

<https://docs.spring.io/spring-cloud-openfeign/docs/current/reference/html/#spring-cloud-feign-overriding-defaults>