si es primera vez, hacer:

git clone https://github.com/javidep/curso-microservicios.git

# Labo RouteFilter

|  |  |
| --- | --- |
| **1** | Bajar fuentes  git pull  curso-microservicios/sesion5/demo-routefilter |
| 2 | * specialroutes-service |
| 3 | Desplegar  cd demo-routefilter  mvn clean package docker:build  docker-compose -f docker/common/docker-compose.yml up |
| 4 | Ejecutar  Get localhost:8910/v1/route/abtesting/licensingservice  Get localhost:8910/v1/route/abtesting/organizationservice |
| 5 | comentar registro insert  modificar   * specialroutes-service/src/main/resources/schema.sql   comentar insert  -- INSERT INTO abtesting (service\_name, active, endpoint, weight) VALUES ('organizationservice', 'Y','http://orgservice-new:8087',5); |
| 6 | Desplegar specialroutes-service  mvn clean package docker:build -f specialroutes-service/pom.xml  docker-compose -f docker/common/docker-compose.yml up specialroutesservice |
| 7 | ejecutar y verificar  http://localhost:5555/api/licensing/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/licenses/f3831f8c-c338-4ebe-a82a-e2fc1d1ff78a |
| 8 | verificar funcionamiento filters en logs |
| 9 | organization-new |
| 10 | modificar especialroute   * specialroutes-service/src/main/resources/schema.sql   descomentar  INSERT INTO abtesting (service\_name, active, endpoint, weight) VALUES ('organizationservice', 'Y','http://orgservice-new:8087',5); |
| 11 | Desplegar specialroutes-service  mvn clean package docker:build -f specialroutes-service/pom.xml  docker-compose -f docker/common/docker-compose.yml up specialroutesservice |
| 12 | ejecutar y verificar  http://localhost:5555/api/licensing/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/licenses/f3831f8c-c338-4ebe-a82a-e2fc1d1ff78a |
| 13 | verificar funcionamiento filters en logs |

## Labo Seguridad

|  |  |
| --- | --- |
| 1 | curso-microservicios/sesion5/demo-security |
| 2 | authentication server |
| 3 | Construir y desplegar  mvn clean package docker:build  docker-compose -f docker/common/docker-compose.yml up |
| 4 | autenticar usuario  Post http://localhost:8901/auth/oauth/token  basic auth   * username/password= eagleeye/thisissecret   body-form data   * grant\_type: password * scope: webclient * username: default.user * password: password1 * grant\_type: password * scope: webclient * username: admin.user * password: password2 |
| 5 | Obtener información de usuario  Get http://localhost:8901/auth/user  header name = Authorization  header value = Bearer <token> |
| 6 | * organization-service |
| 7 | consultar organization sin token  http://localhost:8085/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a |
| 8 | consultar organization con token  http://localhost:8085/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a  header name = Authorization  header value = Bearer <token> |
| 9 | restringir acceso usando roles  modificar   * organization-service/src/main/java/com/thoughtmechanix/organization/security/ResourceServerConfiguration.java   ...  @Override  public void configure(HttpSecurity http) throws Exception{  http  .authorizeRequests()  .antMatchers(HttpMethod.DELETE, "/v1/organizations/\*\*")  .hasRole("ADMIN")  .anyRequest()  .authenticated();  }  ... |
| 10 | desplegar organization  mvn clean package docker:build -f organization-service/pom.xml  docker-compose -f docker/common/docker-compose.yml up organizationservice |
| 11 | probar endpoint delete  delete http://localhost:8085/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a  con token del usuario default.user |
| 12 | probar endpoint delete  delete http://localhost:8085/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a  con token de admin.user |
| 13 | acceder con token de admin.user  http://localhost:5555/api/licensing/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/licenses/f3831f8c-c338-4ebe-a82a-e2fc1d1ff78a  Verificar error 401 en licensing |
| 14 | Propagar token en licensing   * licensing-service/src/main/java/com/thoughtmechanix/licenses/utils/UserContextInterceptor.java   agregar  …  headers.add(UserContext.AUTH\_TOKEN, UserContextHolder.getContext().getAuthToken());  … |
| 15 | desplegar cambios licensing  mvn clean package docker:build -f licensing-service/pom.xml  docker-compose -f docker/common/docker-compose.yml up licensingservice |
| 16 | acceder con token de admin.user  http://localhost:5555/api/licensing/v1/organizations/e254f8c-c442-4ebe-a82a-e2fc1d1ff78a/licenses/f3831f8c-c338-4ebe-a82a-e2fc1d1ff78a |

# Labo seguridad parte 2

|  |  |
| --- | --- |
| 1 | demo-oauth2-frontend |
| 2 | ejecutar authserver  cd authserver  mvn spring-boot:run |
| 3 | test authserver. generar token  curl acme:acmesecret@localhost:9999/uaa/oauth/token \  -d grant\_type=password -d client\_id=acme \  -d username=user -d password=password |
| 4 | ejecutar resource  cd resource  mvn spring-boot:run |
| 5 | ahora, con token  TOKEN=<token>  curl -H "Authorization: Bearer $TOKEN" localhost:9000 |
| 6 | ejecutar ui  cd ui  mvn spring-boot:run |
| 7 | ejecutar en browser  http://localhost:8080/  login con user/password |