ECOLE NORMALE SUPÉRIEURE DE L'ENSEIGNEMENT
TECHNIQUE DE MOHAMMEDIA
UNIVERSITÉ HASSAN II DE CASABLANCA

Département Mathématique Informatique

Systèmes Distribués

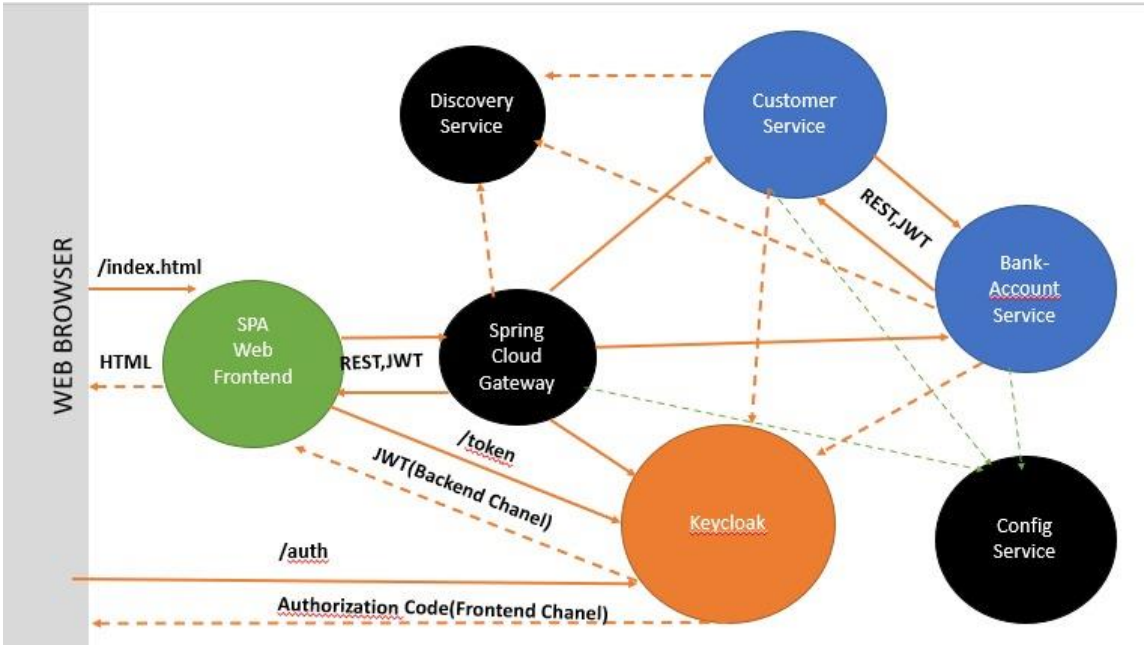
Compte Rendu

Examen Blanc Systèmes Distribués

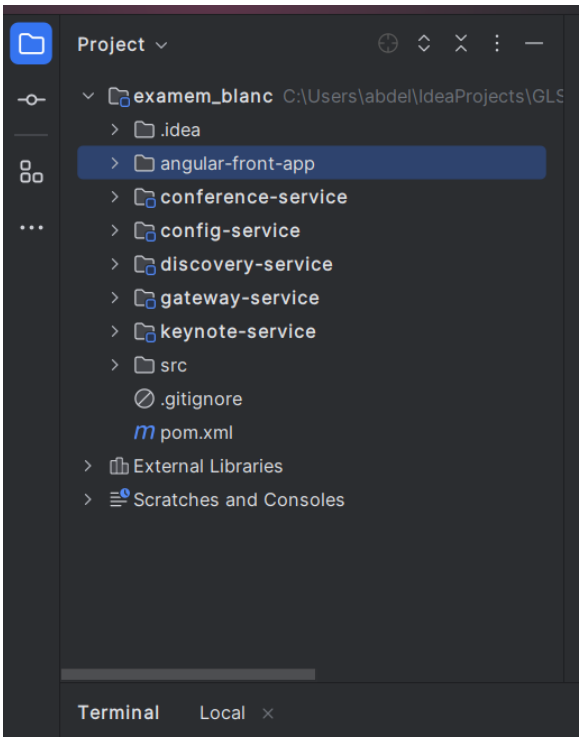
Nom & Prénom : BOUHKKA Abdelilah
GLSID_3

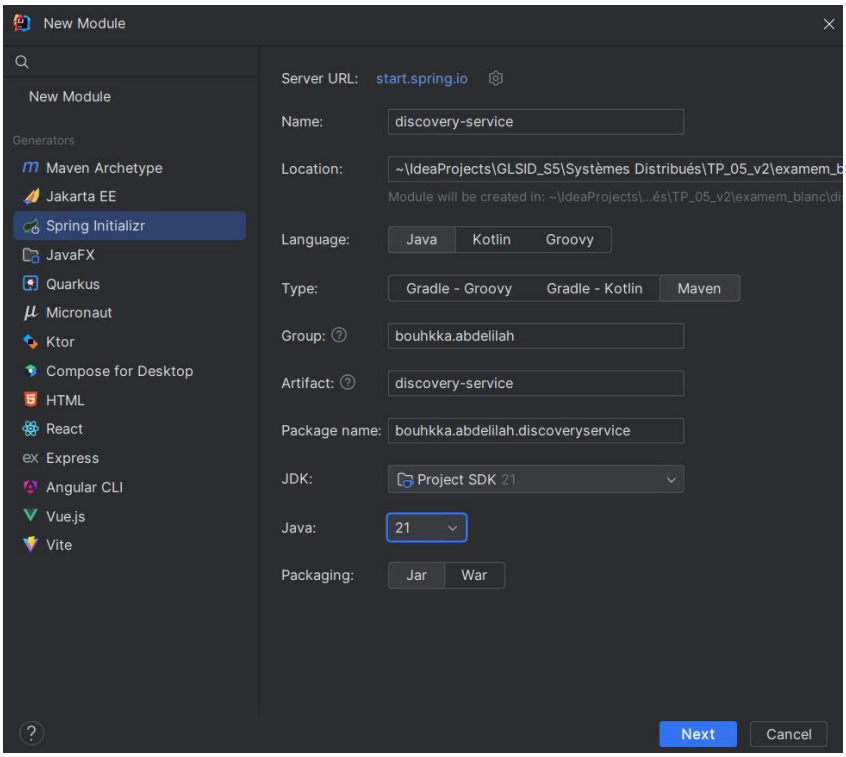
Année universitaire :2024-2025

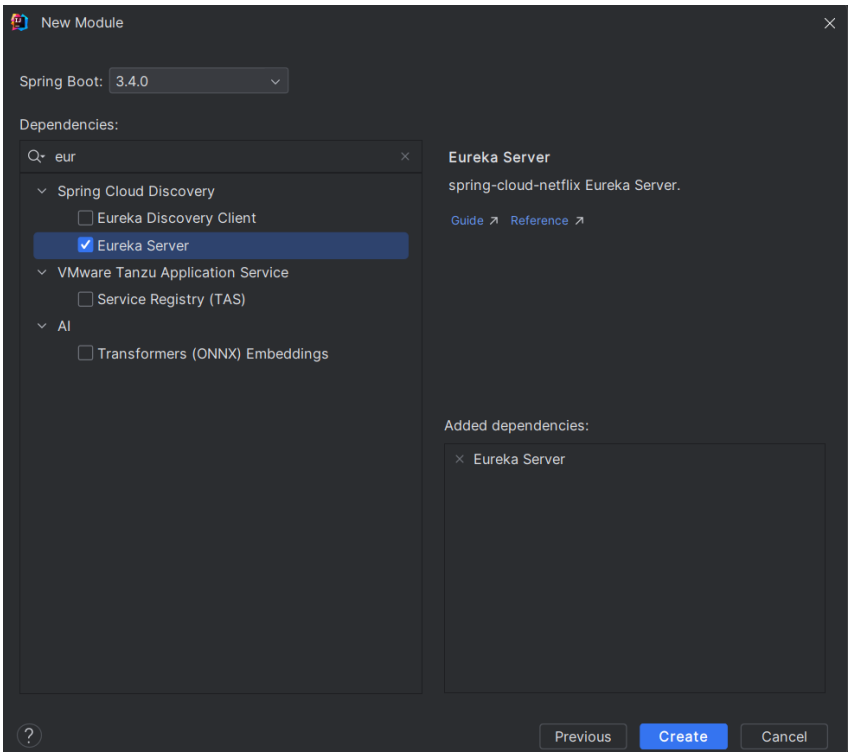
1. **Établir une architecture technique du projet**

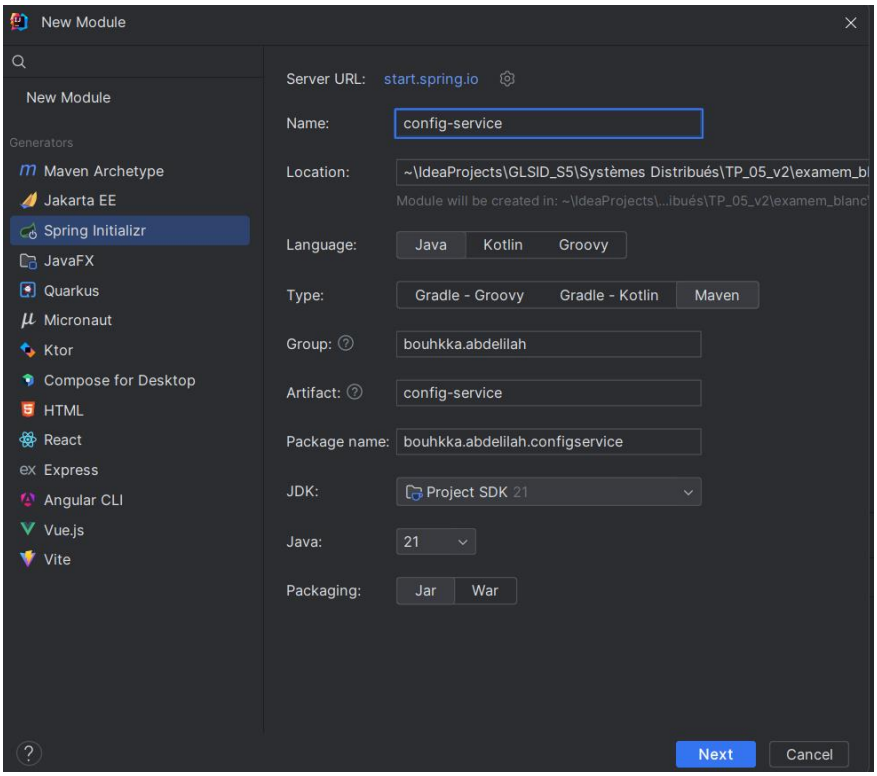


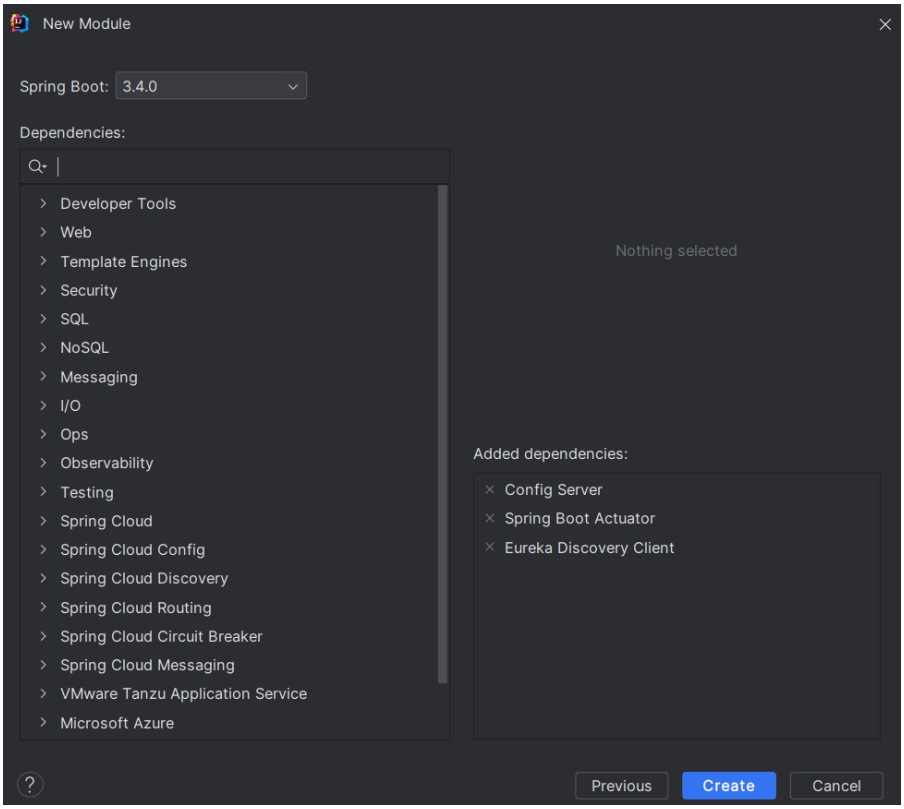
2. **Créer un Projet Maven incluant les micro-services suivants : keynote-service, conference-service, gateway-service, discovery-service, config-service et angular-front-app**

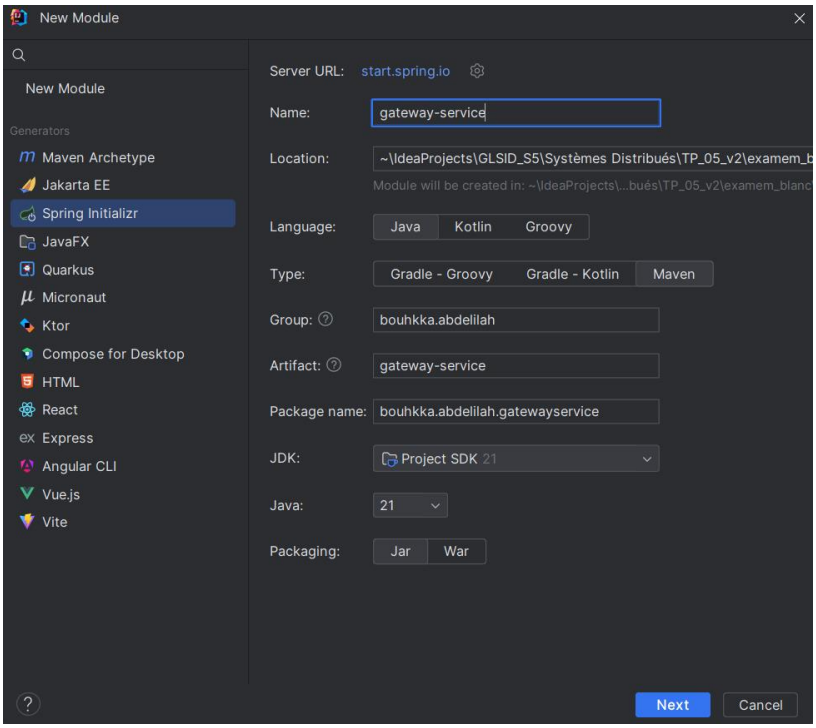


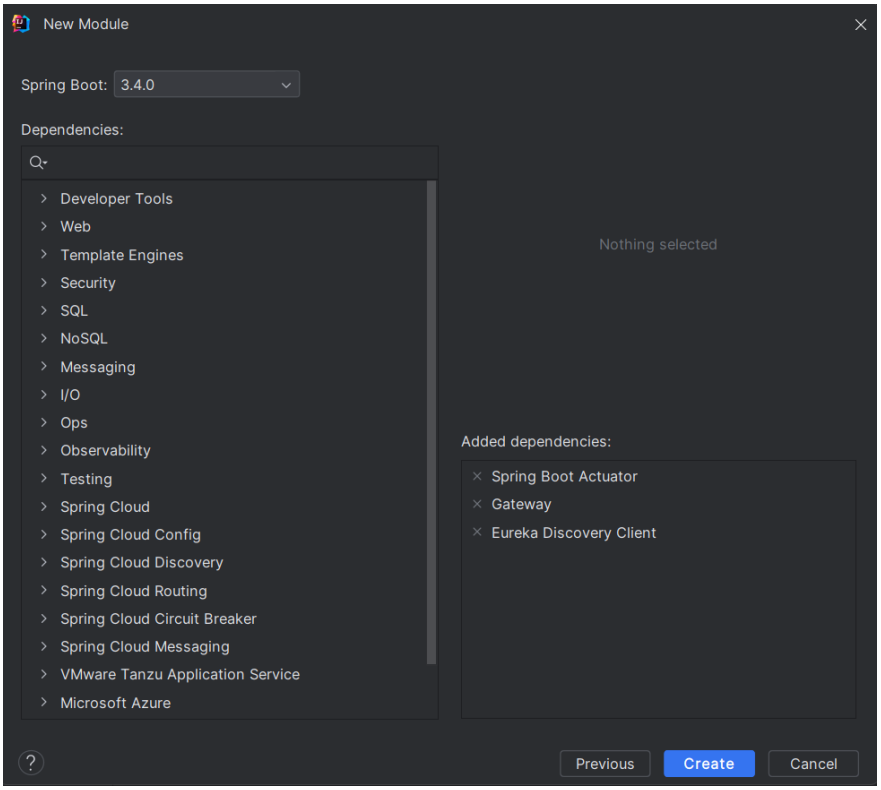

New Module
New Module
Generators
Maven Archetype
Jakarta EE
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Vue.js
Vite
Server URL: start.spring.io
Name: discovery-service
Location: ~\IdeaProjects\GLSID_S5\Systèmes Distribués\TP_05_v2\examem_b
Module will be created in: ~\IdeaProjects\...és\TP_05_v2\examem_blanc\di
Language: Java Kotlin Groovy
Type: Gradle - Groovy Gradle - Kotlin Maven
Group: bouhkka.abdelilah
Artifact: discovery-service
Package name: bouhkka.abdelilah.discoveryservice
JDK: Project SDK 21
Java: 21
Packaging: Jar War
Next
Cancel

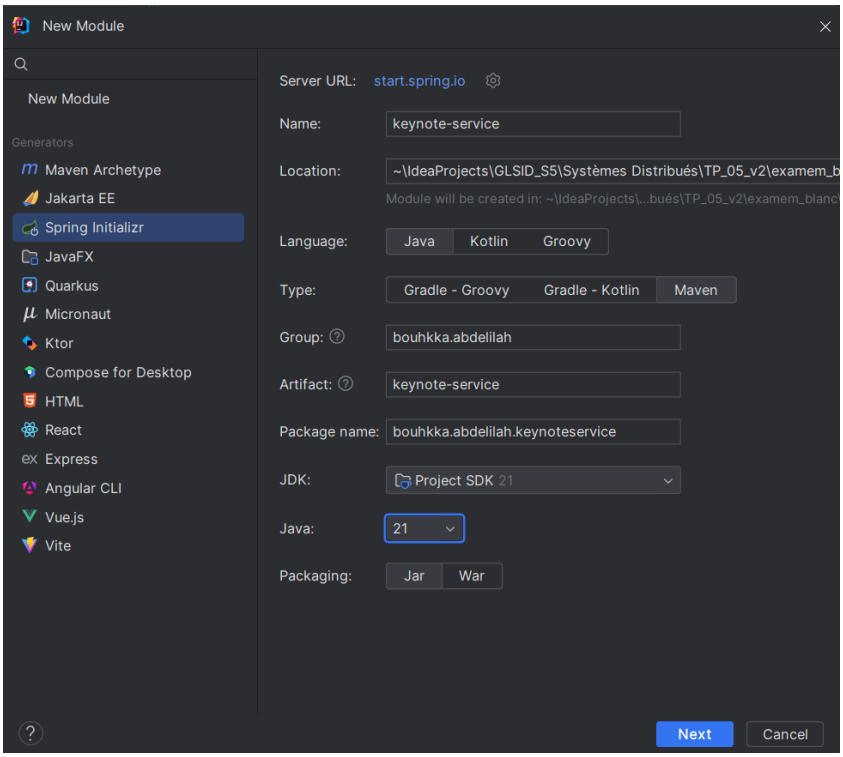

New Module
Spring Boot: 3.4.0
Dependencies:
eur
Spring Cloud Discovery
Eureka Discovery Client
Eureka Server
VMware Tanzu Application Service
Service Registry (TAS)
AI
Transformers (ONNX) Embeddings
Eureka Server
spring-cloud-netflix Eureka Server.
Guide
Reference
Added dependencies:
Eureka Server
Previous
Create
Cancel

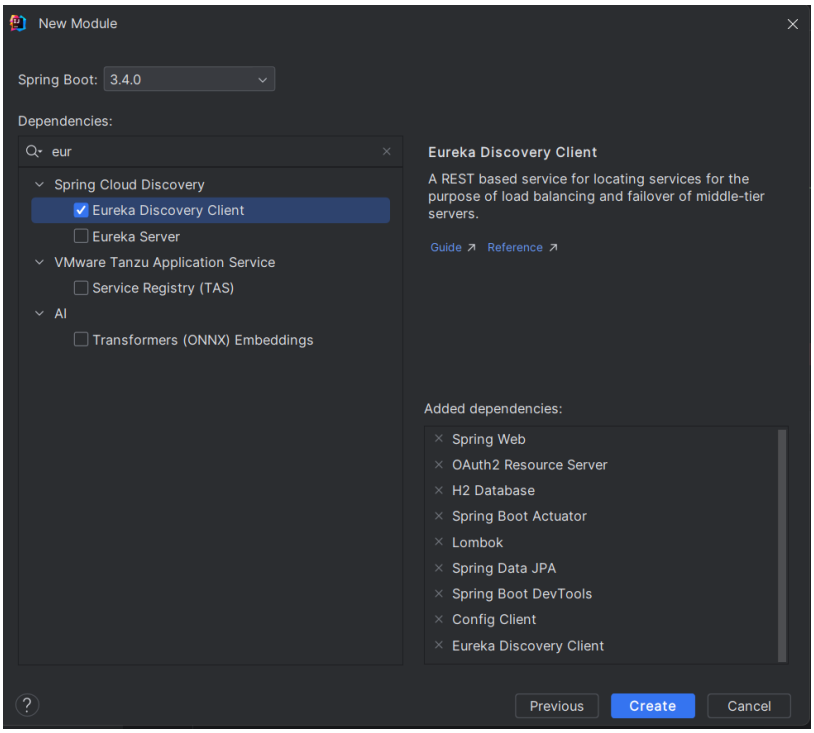

New Module
New Module
Generators
Maven Archetype
Jakarta EE
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Vue.js
Vite
Server URL: start.spring.io
Name: config-service
Location: ~\IdeaProjects\GLSID_S5\Systèmes Distribués\TP_05_v2\examem_bl
Module will be created in: ~\IdeaProjects\...ibués\TP_05_v2\examem_blanc
Language: Java Kotlin Groovy
Type: Gradle - Groovy Gradle - Kotlin Maven
Group: bouhkka.abdelilah
Artifact: config-service
Package name: bouhkka.abdelilah.configservice
JDK: Project SDK 21
Java: 21
Packaging: Jar War
Next
Cancel

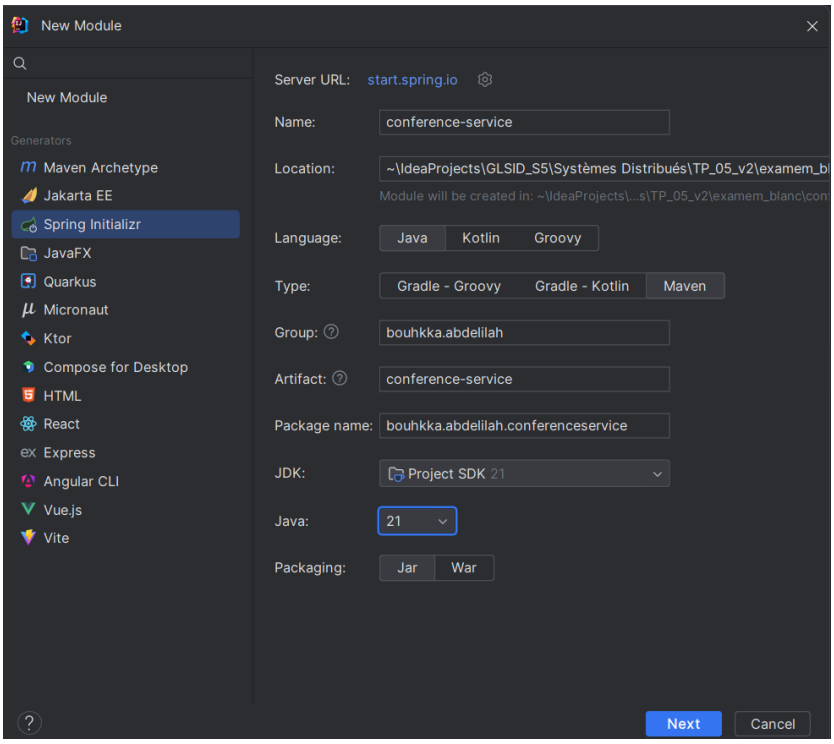

New Module
Spring Boot: 3.4.0
Dependencies:
Developer Tools
Web
Template Engines
Security
SQL
NoSQL
Messaging
I/O
Ops
Observability
Testing
Spring Cloud
Spring Cloud Config
Spring Cloud Discovery
Spring Cloud Routing
Spring Cloud Circuit Breaker
Spring Cloud Messaging
VMware Tanzu Application Service
Microsoft Azure
Nothing selected
Added dependencies:
Config Server
Spring Boot Actuator
Eureka Discovery Client
Previous
Create
Cancel

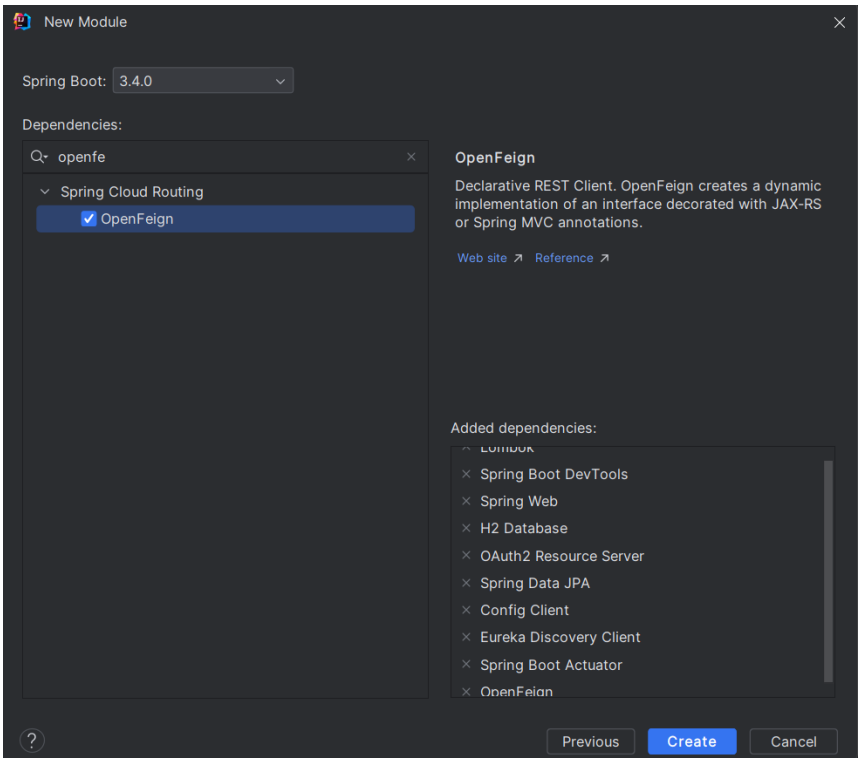

New Module
New Module
Generators
Maven Archetype
Jakarta EE
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Vue.js
Vite
Server URL: start.spring.io
Name: gateway-service
Location: ~\IdeaProjects\GLSID_S5\Systèmes Distribués\TP_05_v2\examem_b
Module will be created in: ~\IdeaProjects\...bués\TP_05_v2\examem_blanc
Language: Java Kotlin Groovy
Type: Gradle - Groovy Gradle - Kotlin Maven
Group: bouhkka.abdelilah
Artifact: gateway-service
Package name: bouhkka.abdelilah.gatewayservice
JDK: Project SDK 21
Java: 21
Packaging: Jar War
Next
Cancel


New Module
Spring Boot: 3.4.0
Dependencies:
Developer Tools
Web
Template Engines
Security
SQL
NoSQL
Messaging
I/O
Ops
Observability
Testing
Spring Cloud
Spring Cloud Config
Spring Cloud Discovery
Spring Cloud Routing
Spring Cloud Circuit Breaker
Spring Cloud Messaging
VMware Tanzu Application Service
Microsoft Azure
Nothing selected
Added dependencies:
Spring Boot Actuator
Gateway
Eureka Discovery Client
Previous
Create
Cancel


New Module
New Module
Generators
Maven Archetype
Jakarta EE
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Vue.js
Vite
Server URL: start.spring.io
Name: keynote-service
Location: ~\IdeaProjects\GLSID_S5\Systèmes Distribués\TP_05_v2\examem_b
Module will be created in: ~\IdeaProjects\...bués\TP_05_v2\examem_blanc
Language: Java Kotlin Groovy
Type: Gradle - Groovy Gradle - Kotlin Maven
Group: bouhkka.abdelilah
Artifact: keynote-service
Package name: bouhkka.abdelilah.keynoteservice
JDK: Project SDK 21
Java: 21
Packaging: Jar War
Next
Cancel


New Module
Spring Boot: 3.4.0
Dependencies:
eur
Spring Cloud Discovery
Eureka Discovery Client
Eureka Server
VMware Tanzu Application Service
Service Registry (TAS)
AI
Transformers (ONNX) Embeddings
Eureka Discovery Client
A REST based service for locating services for the purpose of load balancing and failover of middle-tier servers.
Guide ↗ Reference ↗
Added dependencies:
Spring Web
OAuth2 Resource Server
H2 Database
Spring Boot Actuator
Lombok
Spring Data JPA
Spring Boot DevTools
Config Client
Eureka Discovery Client
Previous
Create
Cancel


New Module
New Module
Generators
Maven Archetype
Jakarta EE
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Vue.js
Vite
Server URL: start.spring.io
Name: conference-service
Location: ~\IdeaProjects\GLSID_S5\Systèmes Distribués\TP_05_v2\examem_bl
Module will be created in: ~\IdeaProjects\...s\TP_05_v2\examem_blanc\con
Language: Java Kotlin Groovy
Type: Gradle - Groovy Gradle - Kotlin Maven
Group: bouhkka.abdelilah
Artifact: conference-service
Package name: bouhkka.abdelilah.conferenceservice
JDK: Project SDK 21
Java: 21
Packaging: Jar War
Next
Cancel


New Module
Spring Boot: 3.4.0
Dependencies:
openfe
Spring Cloud Routing
OpenFeign
OpenFeign
Declarative REST Client. OpenFeign creates a dynamic implementation of an interface decorated with JAX-RS or Spring MVC annotations.
Web site
Reference
Added dependencies:
Spring Boot DevTools
Spring Web
H2 Database
OAuth2 Resource Server
Spring Data JPA
Config Client
Eureka Discovery Client
Spring Boot Actuator
OpenFeign
Previous
Create
Cancel

3. Développer et tester les micro-services discovery-service et gateway-service et config-service

- Gateway-service :

```java
@SpringBootApplication
public class GatewayServiceApplication {

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

    @Bean
    DiscoveryClientRouteDefinitionLocator definitionLocator(
            ReactiveDiscoveryClient rdc,
            DiscoveryLocatorProperties properties){
        return new DiscoveryClientRouteDefinitionLocator(rdc, properties);
    }
}
```

```
spring.application.name=gateway-service

server.port=8888
spring.cloud.discovery.enabled=true
```

```yaml
spring:
  cloud:
    gateway:
      globalcors:
        corsConfigurations:
          '[/**]':
            allowedOrigins: "http://localhost:4200"
            allowedHeaders: "*"
            allowedMethods:
              - GET
              - POST
              - PUT
              - DELETE
```

- discovery-service :

```java
@SpringBootApplication
@EnableEurekaServer
public class DiscoveryServiceApplication {

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

}
```

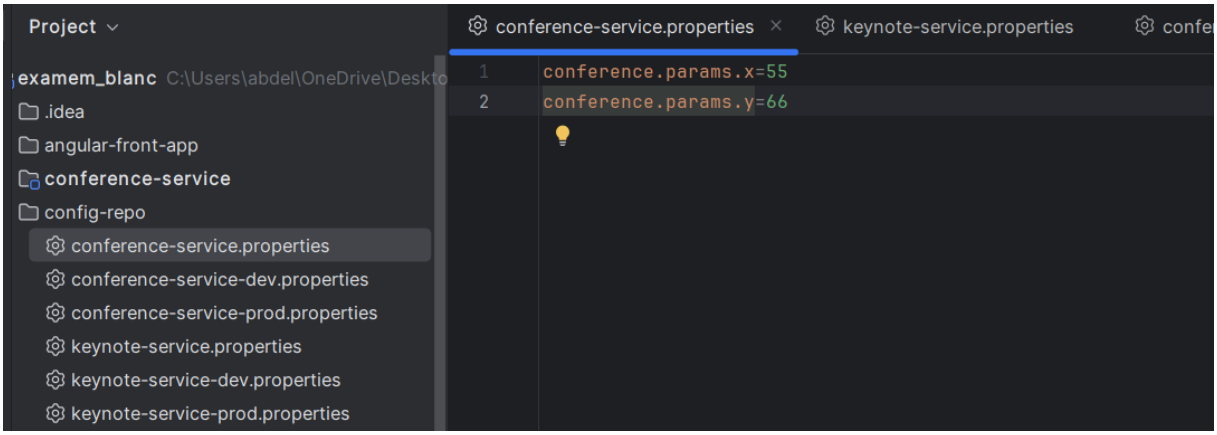
```
spring.application.name=discovery-service

server.port=8761
eureka.client.fetch-registry=false
eureka.client.register-with-eureka=false
```

- config-service :

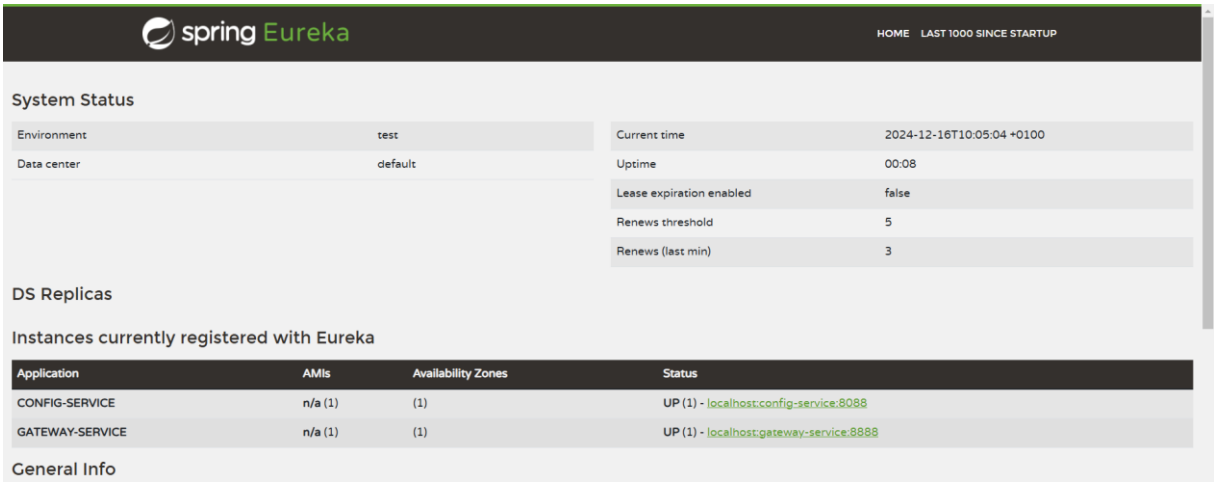
```java
@SpringBootApplication
@EnableConfigServer
@EnableDiscoveryClient
public class ConfigServiceApplication {

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

}
```

```
spring.application.name=config-service

server.port=8088
spring.cloud.config.server.git.uri=file:///C:/Users/abdel/OneDrive/Desktop/GLSID_S5/examem_blanc/config-repo
```

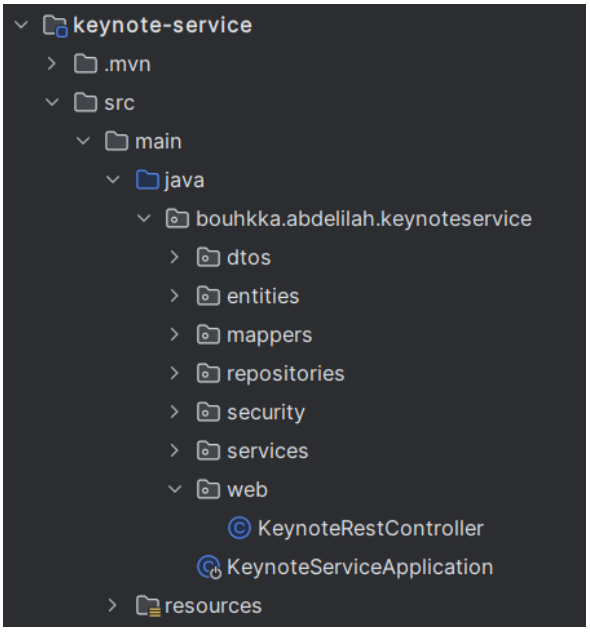
Créer un config-repo qui contient les fichiers de configurations :




```
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc> cd .\config-repo\
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\config-repo> git init
Initialized empty Git repository in C:/Users/abdel/OneDrive/Desktop/GLSID_S5/examem_blanc/config-repo/.git/
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\config-repo> git add .
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\config-repo> git commit -m "first commit"
```



4. **Développer et tester le micro-service Keynote-service (Entities, DAO, service, DTO, Mapper, RestController)**



a. Entities :

```
9 usages
@Entity
@NoArgsConstructor
@AllArgsConstructor
@Getter @Setter @Builder @ToString
public class Keynote {
    @Id
    private String id;
    private String nom;
    private String prenom;
    private String email;
    private String fonction;
}
```

b. Repositoriy

```
import bouhkka.abdelilah.keynoteservice.entities.Keynote;
import org.springframework.data.jpa.repository.JpaRepository;

5 usages
public interface KeynoteRepository extends JpaRepository<Keynote, String> {
}
```

c. DTO :

```java
20 usages
@Data
public class KeynoteDTO {
    private String id;
    private String nom;
    private String prenom;
    private String email;
    private String fonction;
}
```

Ajouter les dépendances de **mapstruct** et **springdoc openAPI**

```xml
        <dependency>
            <groupId>org.springdoc</groupId>
            <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
            <version>2.3.0</version>
        </dependency>
        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct</artifactId>
            <version>1.5.0.Final</version>
        </dependency>
        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct-processor</artifactId>
            <version>1.5.0.Final</version>
            <scope>provided</scope>
        </dependency>
```

d. Mapper :

```java
3 usages  1 implementation
@Mapper
public interface KeynoteMapper {
    no usages
    KeynoteMapper INSTANCE = Mappers.getMapper(KeynoteMapper.class);
    no usages  1 implementation
    KeynoteDTO keynoteToKeynoteDTO(Keynote keynote);
    no usages  1 implementation
    Keynote keynoteDTOToKeynote(KeynoteDTO keynoteDTO);
}
```

e. Keynote service :

```java
4 usages  1 implementation
public interface KeynoteService {
    1 usage  1 implementation
    List<KeynoteDTO> getAllKeynotes();
    2 usages  1 implementation
    Optional<KeynoteDTO> getKeynoteById(String id);
    1 usage  1 implementation
    KeynoteDTO createKeynote(KeynoteDTO keynoteDTO);
    1 usage  1 implementation
    void deleteKeynote(String id);
    1 usage  1 implementation
    KeynoteDTO updateKeynote(KeynoteDTO keynoteDTO);
}
```

f. Keynote service implémentation :

```java
@Service
@Transactional
public class KeynoteServiceImpl implements KeynoteService{

    @Autowired
    private KeynoteRepository keynoteRepository;
    no usages
    @Override
    public List<KeynoteDTO> getAllKeynotes() {
        List<Keynote> keynotes = keynoteRepository.findAll();
        return keynotes.stream() Stream<Keynote>
                .map(KeynoteMapper.INSTANCE::keynoteToKeynoteDTO) Stream<KeynoteDTO>
                .toList();
    }

    no usages
    @Override
    public Optional<KeynoteDTO> getKeynoteById(String id) {
        return keynoteRepository.findById(id)
                .map(KeynoteMapper.INSTANCE::keynoteToKeynoteDTO);
    }
```

```java
    @Override
    public KeynoteDTO createKeynote(KeynoteDTO keynoteDTO) {
        Keynote keynote = KeynoteMapper.INSTANCE.keynoteDTOToKeynote(keynoteDTO);
        Keynote savedKeynote = keynoteRepository.save(keynote);
        return KeynoteMapper.INSTANCE.keynoteToKeynoteDTO(savedKeynote);
    }

    no usages
    @Override
    public void deleteKeynote(String id) {
        keynoteRepository.deleteById(id);
    }
}
```

g. Controller

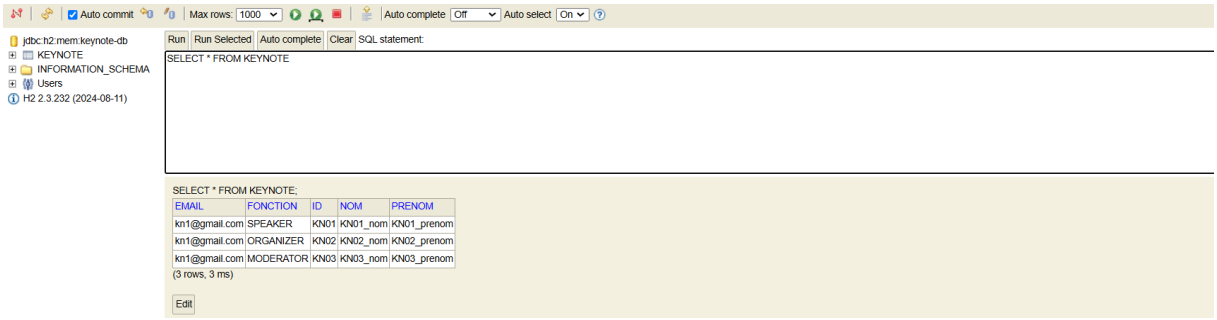
```java
@RestController
@RequestMapping("/api")
public class KeynoteRestController {

    7 usages
    private KeynoteService keynoteService;

    public KeynoteRestController(KeynoteService keynoteService) {
        this.keynoteService = keynoteService;
    }

    @GetMapping("/keynotes")
    public List<KeynoteDTO> keynoteList(){
        return keynoteService.getAllKeynotes();
    }

    @GetMapping("/keynotes/{id}")
    public KeynoteDTO keynoteById(@PathVariable String id){
        return keynoteService.getKeynoteById(id).get();
    }
```

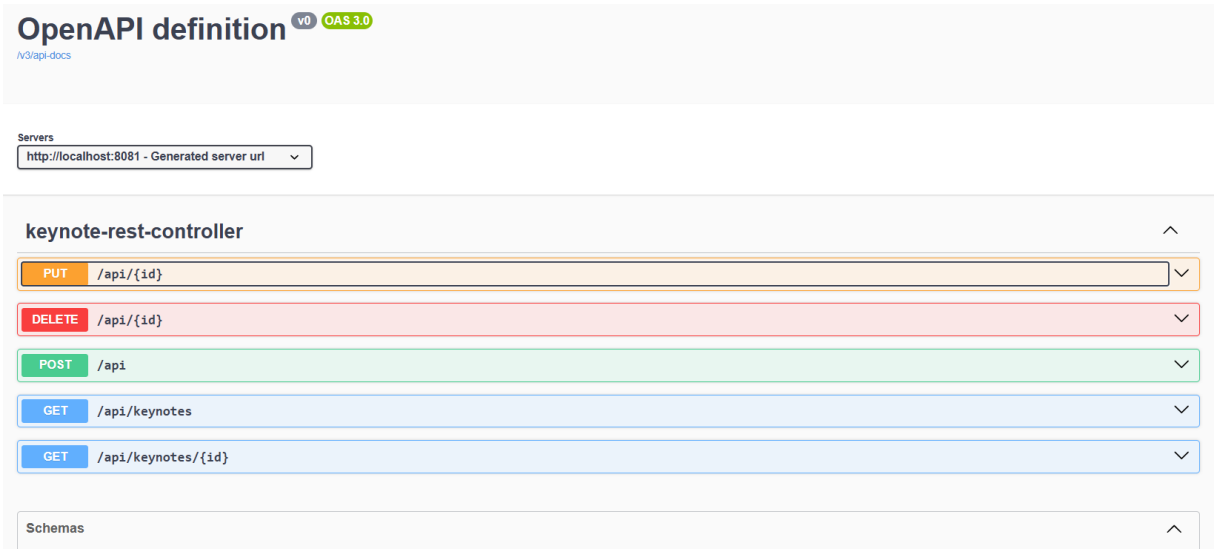
```java
@PostMapping
public KeynoteDTO createKeynote(@RequestBody KeynoteDTO keynoteDTO) {
    KeynoteDTO savedKeynote = keynoteService.createKeynote(keynoteDTO);
    return savedKeynote;
}

@PutMapping("/{id}")
public KeynoteDTO updateKeynote(@PathVariable String id, @RequestBody KeynoteDTO keynoteDTO) {
    if (!keynoteService.getKeynoteById(id).isPresent()) {
        return null;
    }
    keynoteDTO.setId(id);  // Make sure the ID is set
    KeynoteDTO updatedKeynote = keynoteService.updateKeynote(keynoteDTO);
    return updatedKeynote;
}

@DeleteMapping("/{id}")
public void deleteKeynote(@PathVariable String id) {
    keynoteService.deleteKeynote(id);
}
```

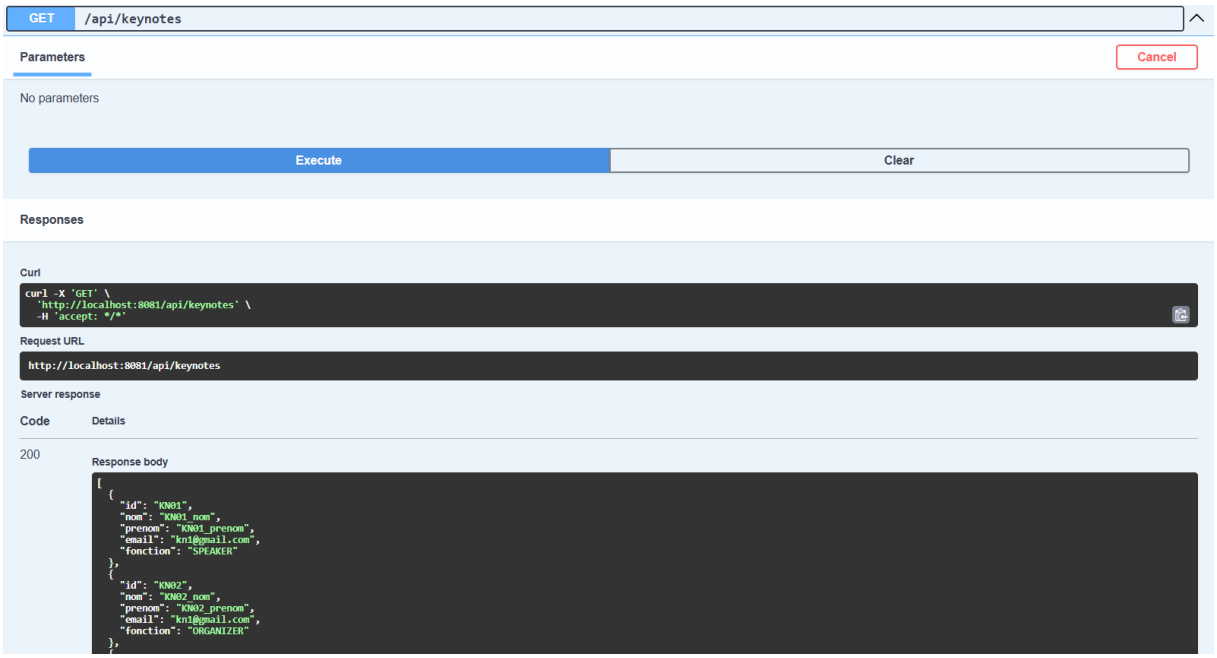
h. Application.properties :

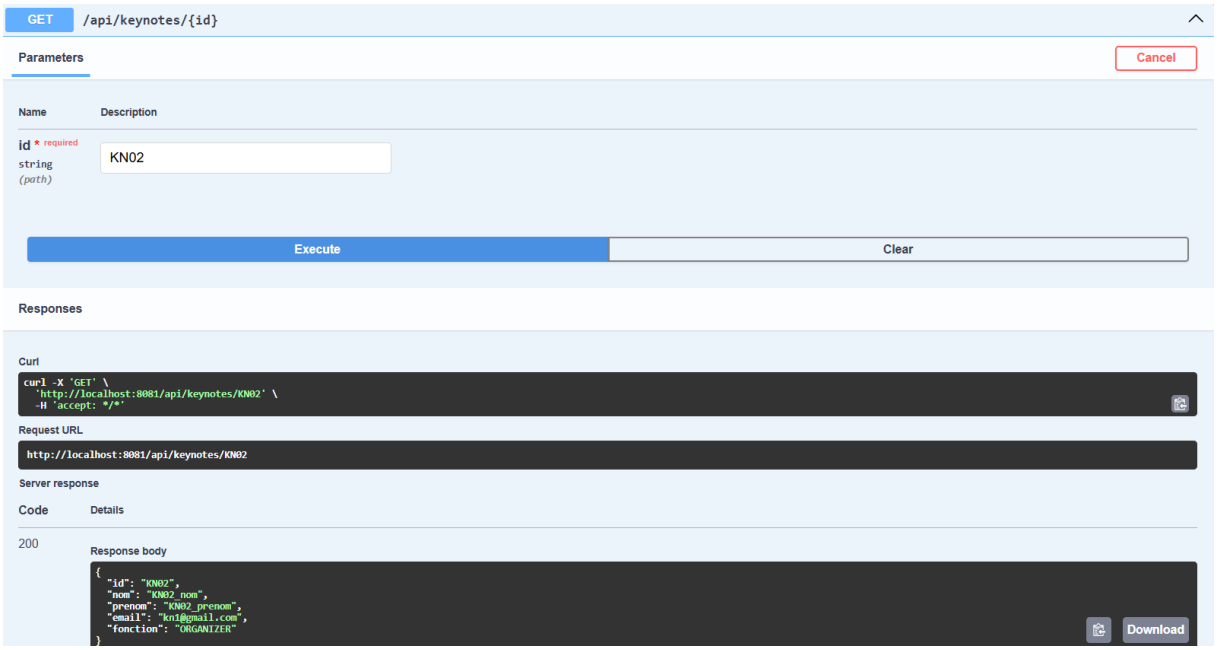
```
spring.application.name=keynote-service

server.port=8081
spring.datasource.url=jdbc:h2:mem:keynote-db
spring.h2.console.enabled=true

spring.cloud.config.enabled=false
spring.cloud.discovery.enabled=false
```

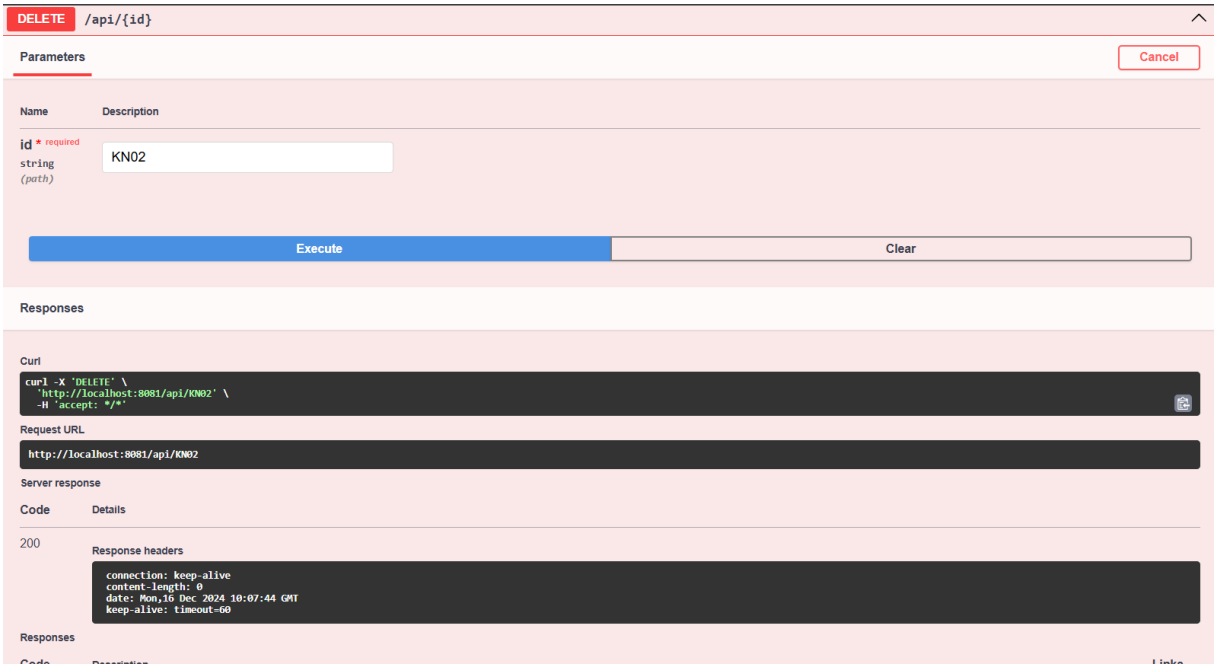
Test :



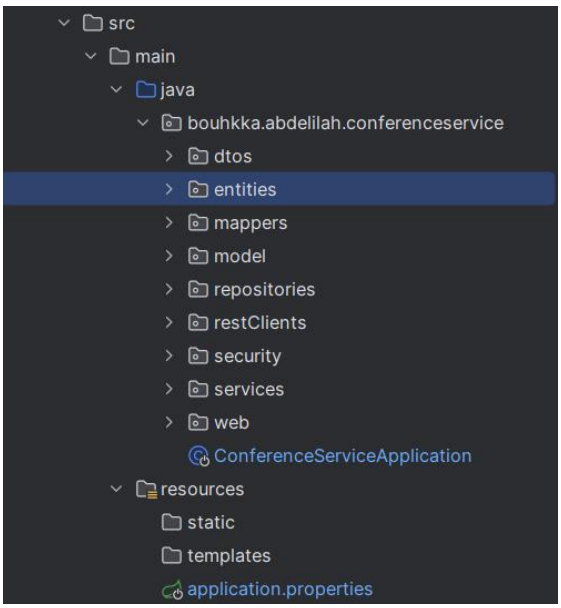



GET
/api/keynotes
Parameters
Cancel
No parameters
Execute
Clear
Responses
Curl
curl -X 'GET' \
'http://localhost:8081/api/keynotes' \
-H 'accept: */*'
Request URL
http://localhost:8081/api/keynotes
Server response
Code
Details
200
Response body
[
{
"id": "KN01",
"nom": "KN01_nom",
"prenom": "KN01_prenom",
"email": "kn1@gmail.com",
"fonction": "SPEAKER"
},
{
"id": "KN02",
"nom": "KN02_nom",
"prenom": "KN02_prenom",
"email": "kn1@gmail.com",
"fonction": "ORGANIZER"
},


GET
/api/keynotes/{id}
Parameters
Cancel
Name
Description
id * required
string
(path)
KN02
Execute
Clear
Responses
Curl
curl -X 'GET' \
'http://localhost:8081/api/keynotes/KN02' \
-H 'accept: */*'
Request URL
http://localhost:8081/api/keynotes/KN02
Server response
Code
Details
200
Response body
{
"id": "KN02",
"nom": "KN02_nom",
"prenom": "KN02_prenom",
"email": "kn1@gmail.com",
"fonction": "ORGANIZER"
}
Download


DELETE
/api/{id}
Parameters
Cancel
Name
Description
id * required
string
(path)
KN02
Execute
Clear
Responses
Curl
curl -X 'DELETE' \
'http://localhost:8081/api/KN02' \
-H 'accept: */*'
Request URL
http://localhost:8081/api/KN02
Server response
Code
Details
200
Response headers
connection: keep-alive
content-length: 0
date: Mon,16 Dec 2024 10:07:44 GMT
keep-alive: timeout=60
Responses

5. **Développer et tester le micro-service conférence-service (Entities, DAO, service, DTO, Mapper, RestController, Client Rest Open Feign)**



a. Entities

```java
@Entity
@NoArgsConstructor
@AllArgsConstructor
@Getter @Setter @Builder @ToString
public class Conference {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String titre;
    private String type;
    private LocalDate date;
    private int duree;
    private int nombreInscrits;
    private double score;

    @Transient
    private List<Keynote> keynotes;
}
```

b. DTO

```java
31 usages
@Data
public class ConferenceDTO {
    private Long id;
    private String titre;
    private String type;
    private LocalDate date;
    private int duree;
    private int nombreInscrits;
    private double score;
    private List<Keynote> keynotes;
}
```

c. Model

```java
@NoArgsConstructor
@AllArgsConstructor
@Getter
@Setter
@Builder
@ToString
public class Keynote {
    private String id;
    private String nom;
    private String prenom;
    private String email;
    private String fonction;
}
```

d. Mappers

```java
8 usages  1 implementation
@Mapper
public interface ConferenceMapper {
    4 usages
    ConferenceMapper INSTANCE = Mappers.getMapper(ConferenceMapper.class);
    3 usages  1 implementation
    ConferenceDTO conferenceToConferenceDTO(Conference conference);
    1 usage  1 implementation
    Conference conferenceDTOToConference(ConferenceDTO conferenceDTO);
}
```

e. Repository

```java
4 usages
public interface ConferenceRepository extends JpaRepository<Conference, String> {
}
```

f. RestClient

```java
4 usages
@FeignClient(url = "http://localhost:8081", name = "keynote-service")
public interface KeynoteRestClient {
    @GetMapping("/api/keynotes")
    List<Keynote> getAllKeynotes();
    @GetMapping("/api/keynotes/{id}")
    Keynote findKeynoteById(@PathVariable String id);
}
```

g. ConferenceServices

```java
4 usages  1 implementation
public interface ConferenceService {
    1 usage  1 implementation
    List<ConferenceDTO> getAllConferences();
    2 usages  1 implementation
    Optional<ConferenceDTO> getConferenceById(Long id);
    1 usage  1 implementation
    ConferenceDTO createConference(ConferenceDTO conferenceDTO);
    1 usage  1 implementation
    void deleteConference(Long id);
    1 usage  1 implementation
    ConferenceDTO updateConference(ConferenceDTO conferenceDTO);
}
```

h. ConferenceServiceImpl :

```java
@Service
@Transactional
public class ConferenceServiceImpl implements ConferenceService {

    @Autowired
    private ConferenceRepository conferenceRepository;
    1 usage
    @Override
    public List<ConferenceDTO> getAllConferences() {
        List<Conference> conferences = conferenceRepository.findAll();
        return conferences.stream() Stream<Conference>
                .map(ConferenceMapper.INSTANCE::conferenceToConferenceDTO) Stream<ConferenceDTO>
                .toList();
    }

    2 usages
    @Override
    public Optional<ConferenceDTO> getConferenceById(Long id) {
        return conferenceRepository.findById(String.valueOf(id))
                .map(ConferenceMapper.INSTANCE::conferenceToConferenceDTO);
    }
```

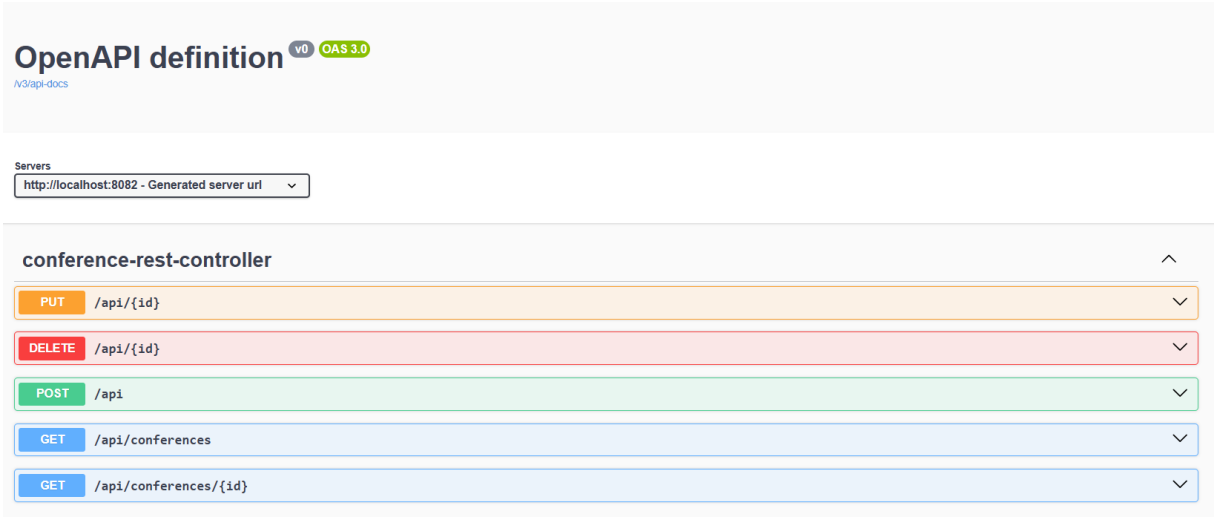
```java
    1 usage
    @Override
    public ConferenceDTO createConference(ConferenceDTO conferenceDTO) {
        Conference conference = ConferenceMapper.INSTANCE.conferenceDTOToConference(conferenceDTO);
        Conference savedConference = conferenceRepository.save(conference);
        return ConferenceMapper.INSTANCE.conferenceToConferenceDTO(savedConference);
    }

    1 usage
    @Override
    public void deleteConference(Long id) { conferenceRepository.deleteById(String.valueOf(id)); }

    1 usage
    @Override
    public ConferenceDTO updateConference(ConferenceDTO conferenceDTO) { return null; }
}
```

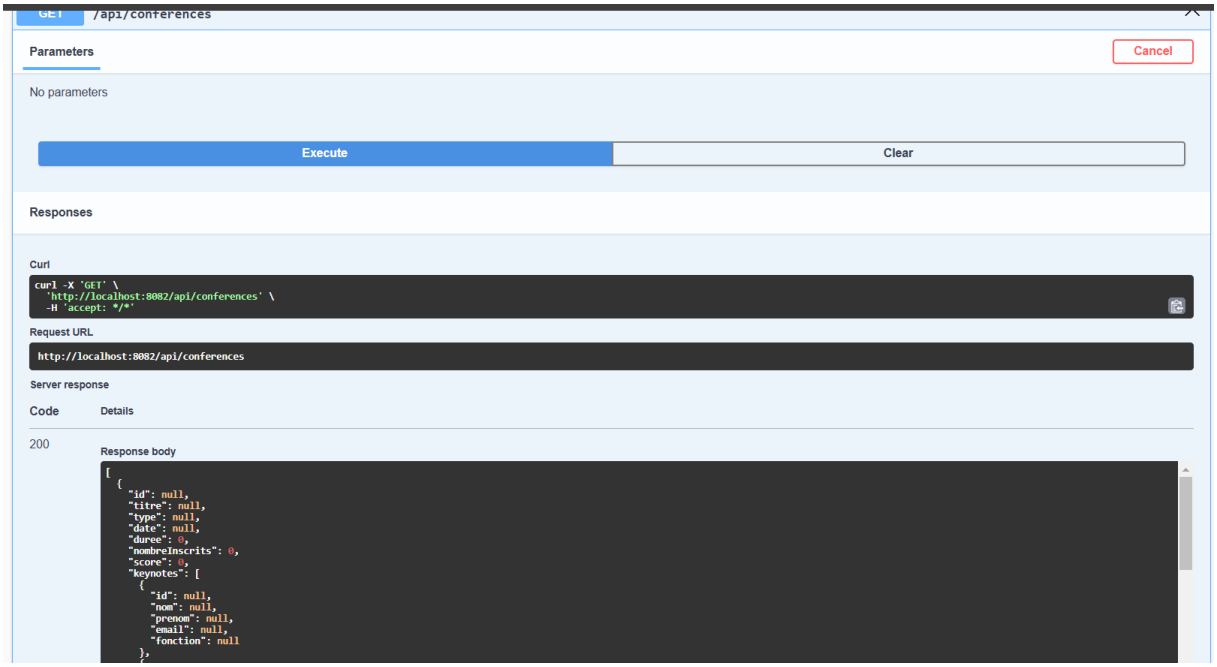
i. Controllers

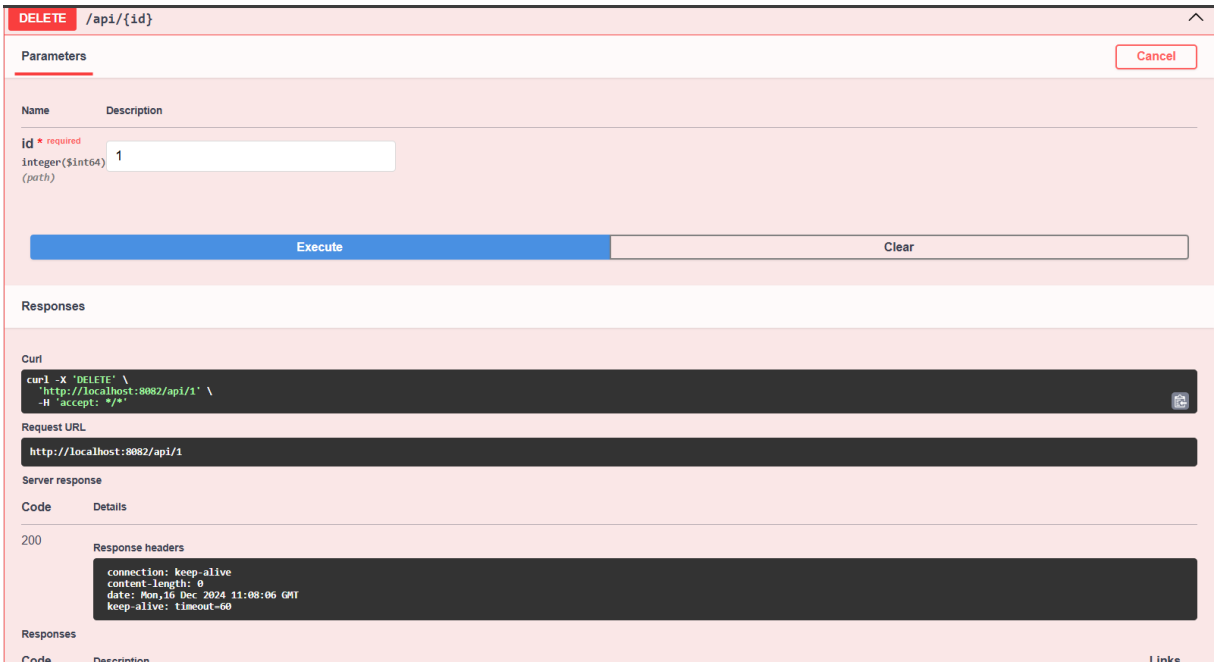
```java
@RestController
@RequestMapping("/api")
public class ConferenceRestController {

    7 usages
    private ConferenceService conferenceService;
    2 usages
    private KeynoteRestClient keynoteRestClient;

    public ConferenceRestController(ConferenceService conferenceService, KeynoteRestClient keynoteRestClient) {
        this.conferenceService = conferenceService;
        this.keynoteRestClient = keynoteRestClient;
    }

    @GetMapping("/conferences")
    public List<ConferenceDTO> conferenceList(){
        List<ConferenceDTO> conferences = conferenceService.getAllConferences();
        conferences.forEach(c->{
                c.setKeynotes(keynoteRestClient.getAllKeynotes());
        });
        return conferences;
    }

    @GetMapping("/conferences/{id}")
    public ConferenceDTO conferenceById(@PathVariable Long id) { return conferenceService.getConferenceById(id).get(); }
```

```java
    @PostMapping
    public ConferenceDTO createConference(@RequestBody ConferenceDTO conferenceDTO) {
        ConferenceDTO savedConference = conferenceService.createConference(conferenceDTO);
        return savedConference;
    }

    @PutMapping("/{id}")
    public ConferenceDTO updateConference(@PathVariable Long id, @RequestBody ConferenceDTO conferenceDTO) {
        if (!conferenceService.getConferenceById(id).isPresent()) {
            return null;
        }
        conferenceDTO.setId(id);  // Make sure the ID is set
        ConferenceDTO updatedConference = conferenceService.updateConference(conferenceDTO);
        return updatedConference;
    }

    @DeleteMapping("/{id}")
    public void deleteConference(@PathVariable Long id) { conferenceService.deleteConference(id); }
```

j. Testing :


OpenAPI definition
v0
OAS 3.0
/v3/api-docs
Servers
http://localhost:8082 - Generated server url
conference-rest-controller
PUT /api/{id}
DELETE /api/{id}
POST /api
GET /api/conferences
GET /api/conferences/{id}


GET /api/conferences
Parameters
Cancel
No parameters
Execute
Clear
Responses
Curl
curl -X 'GET' \
'http://localhost:8082/api/conferences' \
-H 'accept: */*'
Request URL
http://localhost:8082/api/conferences
Server response
Code
Details
200
Response body
[
{
"id": null,
"titre": null,
"type": null,
"date": null,
"duree": 0,
"nombreInscrits": 0,
"score": 0,
"keynotes": [
{
"id": null,
"nom": null,
"prenom": null,
"email": null,
"fonction": null
},

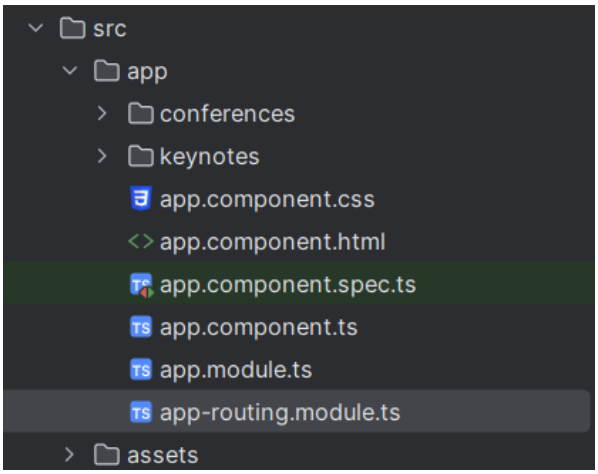

DELETE
/api/{id}
Parameters
Cancel
Name
Description
id * required
integer($int64)
(path)
1
Execute
Clear
Responses
Curl
curl -X 'DELETE' \
'http://localhost:8082/api/1' \
-H 'accept: */*'
Request URL
http://localhost:8082/api/1
Server response
Code
Details
200
Response headers
connection: keep-alive
content-length: 0
date: Mon,16 Dec 2024 11:08:06 GMT
keep-alive: timeout=60
Responses

6. Développer un simple frontend web pour l'application

```
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\angular-front-app> npm i bootstrap bootstrap-icons

added 3 packages, and audited 900 packages in 5s

126 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\angular-front-app>
```

```
        "styles": [
          "src/styles.css",
          "node_modules/bootstrap/dist/css/bootstrap.min.css"
        ],
        "scripts": [
          "node_modules/bootstrap/dist/js/bootstrap.bundle.js"
        ]
      },
```

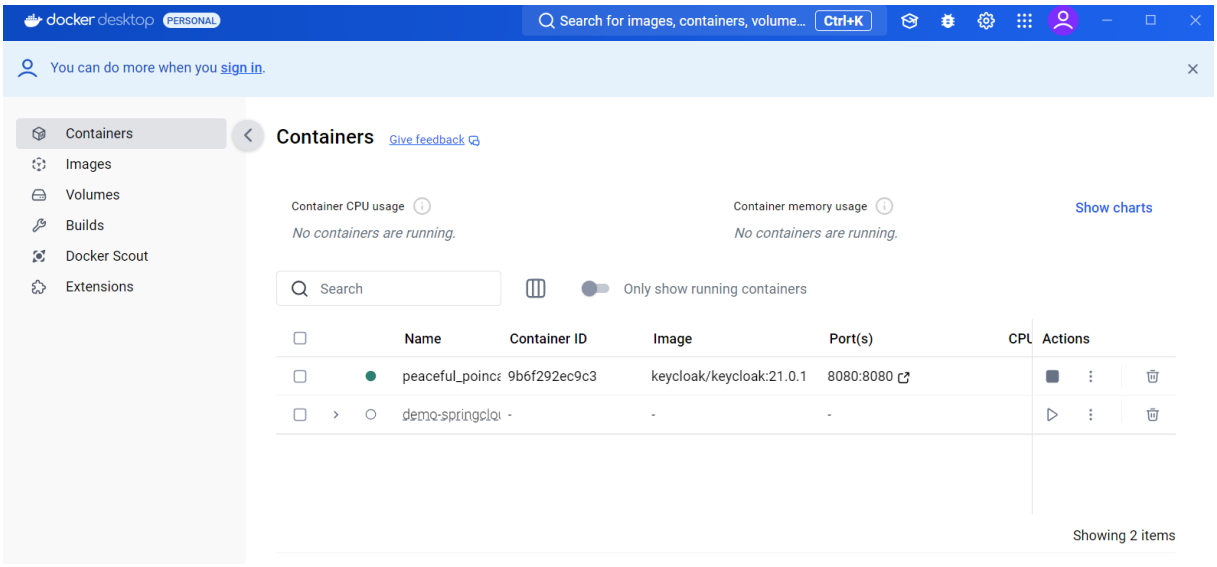
```
const routes: Routes = [
  {path : "keynotes", component : KeynotesComponent},
  {path : "conferences", component : ConferencesComponent}
];
```



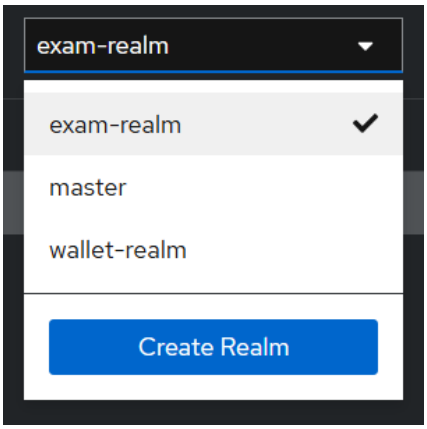
7. Sécuriser l'application avec une authentification Keycloak

Demarer Docker Desctop Et demarer Keycloak .

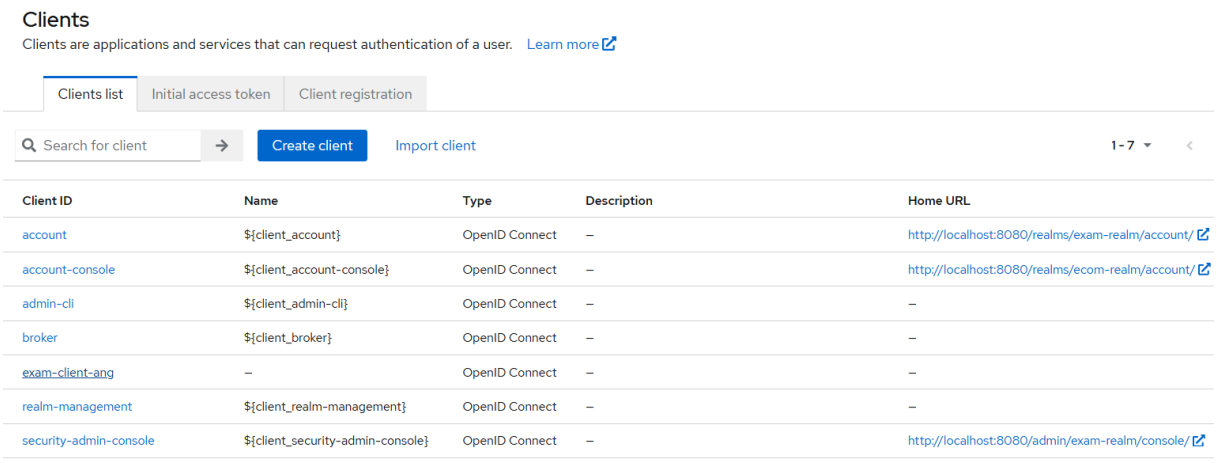
```
C:\Users\abdel>docker run -p 8080:8080 -e KEYCLOAK_ADMIN=admin -e KEYCLOAK_ADMIN_PASSWORD=admin quay.io/keycloak/keycloa
k:21.0.1 start-dev
Updating the configuration and installing your custom providers, if any. Please wait.
```



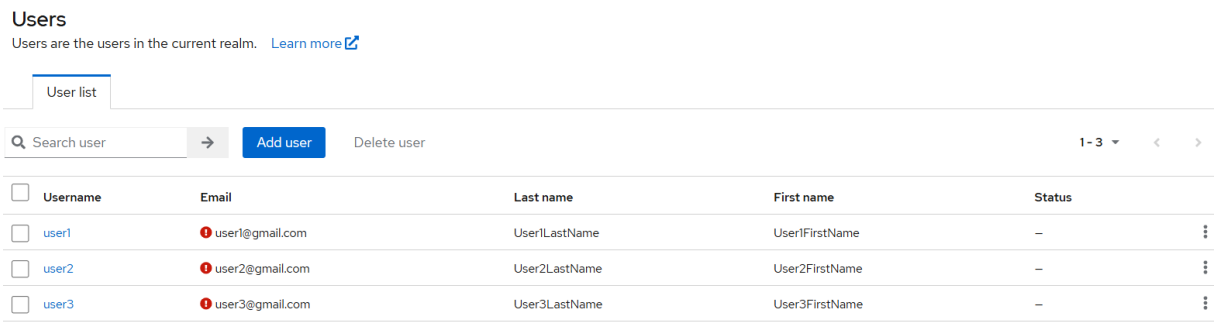
k. Creation d'un nouveau realm



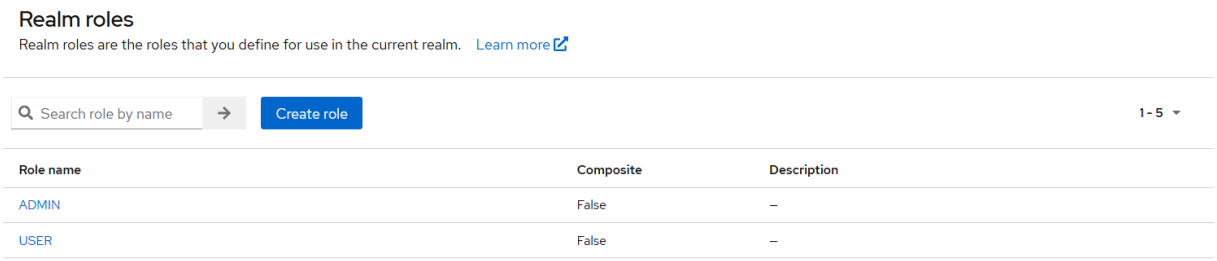
Creation de client



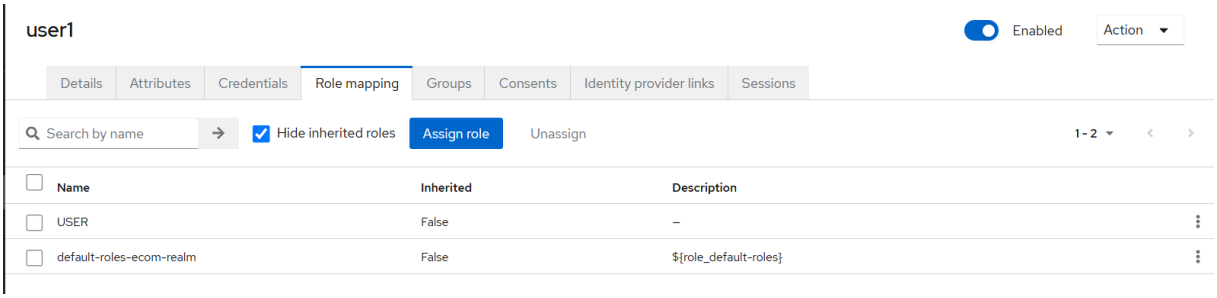
Creation des utilisateurs



Creation des roles



Affection des roles aux utilisateurs



- Cote backend :

Créer JWTAuthConverter et securityConfig pour les micro services

```java
@Component
public class JwtAuthConverter implements Converter<Jwt, AbstractAuthenticationToken> {
    1 usage
    private final JwtGrantedAuthoritiesConverter jwtGrantedAuthoritiesConverter=new JwtGrantedAuthoritiesConverter();
    @Override
    public AbstractAuthenticationToken convert(Jwt jwt) {
        Collection<GrantedAuthority> authorities = Stream.concat(
                jwtGrantedAuthoritiesConverter.convert(jwt).stream(),
                extractResourceRoles(jwt).stream()
        ).collect(Collectors.toSet());
        return new JwtAuthenticationToken(jwt, authorities,jwt.getClaim("preferred_username"));
    }
    1 usage
    private Collection<GrantedAuthority> extractResourceRoles(Jwt jwt) {
        Map<String , Object> realmAccess;
        Collection<String> roles;
        if(jwt.getClaim("realm_access")==null){
            return Set.of();
        }
        realmAccess = jwt.getClaim("realm_access");
        roles = (Collection<String>) realmAccess.get("roles");
        return roles.stream().map(role->new SimpleGrantedAuthority(role)).collect(Collectors.toSet());
    }

}
```

```java
@Configuration
@EnableWebSecurity
@EnableMethodSecurity(prePostEnabled = true)
public class SecurityConfig {
    2 usages
    private JwtAuthConverter jwtAuthConverter;

    public SecurityConfig(JwtAuthConverter jwtAuthConverter) {
        this.jwtAuthConverter = jwtAuthConverter;
    }

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity httpSecurity) throws Exception {
        return httpSecurity
                .cors(Customizer.withDefaults())
                .sessionManagement(sm->sm.sessionCreationPolicy(SessionCreationPolicy.STATELESS))
                .csrf(csrf->csrf.disable())
                .headers(h->h.frameOptions(fo->fo.disable()))
                .authorizeHttpRequests(ar->ar.requestMatchers("/h2-console/**").permitAll())
                .authorizeHttpRequests(ar->ar.requestMatchers("/api/conferences/**", "/swagger-ui.html","/v3/**","/swa
                .authorizeHttpRequests(ar->ar.anyRequest().authenticated())
                .oauth2ResourceServer(o2->o2.jwt(jwt->jwt.jwtAuthenticationConverter(jwtAuthConverter)))
                //.oauth2ResourceServer(o2->o2.jwt(Customizer.withDefaults()))
                .build();
    }
```

```java
    @Bean
    CorsConfigurationSource corsConfigurationSource() {
        CorsConfiguration configuration = new CorsConfiguration();
        configuration.setAllowedOrigins(Arrays.asList("*"));
        configuration.setAllowedMethods(Arrays.asList("*"));
        configuration.setAllowedHeaders(Arrays.asList("*"));
        configuration.setExposedHeaders(Arrays.asList("*"));
        UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();
        source.registerCorsConfiguration( pattern: "/**", configuration);
        return source;
    }
}
```

Ajouter intercepteur au service de conference

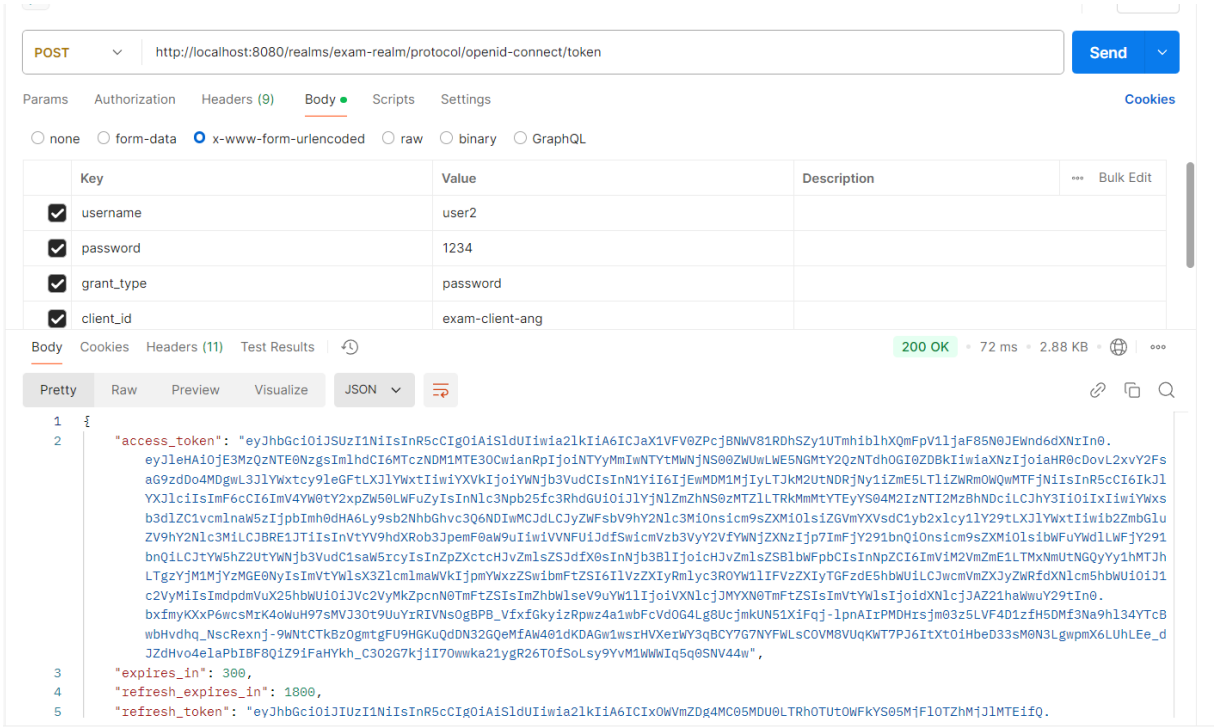
```java
@Component
public class FeignInterceptor implements RequestInterceptor {
    @Override
    public void apply(RequestTemplate requestTemplate) {
        SecurityContext context = SecurityContextHolder.getContext();
        Authentication authentication = context.getAuthentication();
        JwtAuthenticationToken jwtAuthenticationToken= (JwtAuthenticationToken) authentication;
        String jwtAccessToken = jwtAuthenticationToken.getToken().getTokenValue();
        requestTemplate.header( name: "Authorization", ...values: "Bearer "+jwtAccessToken);
    }
}
```

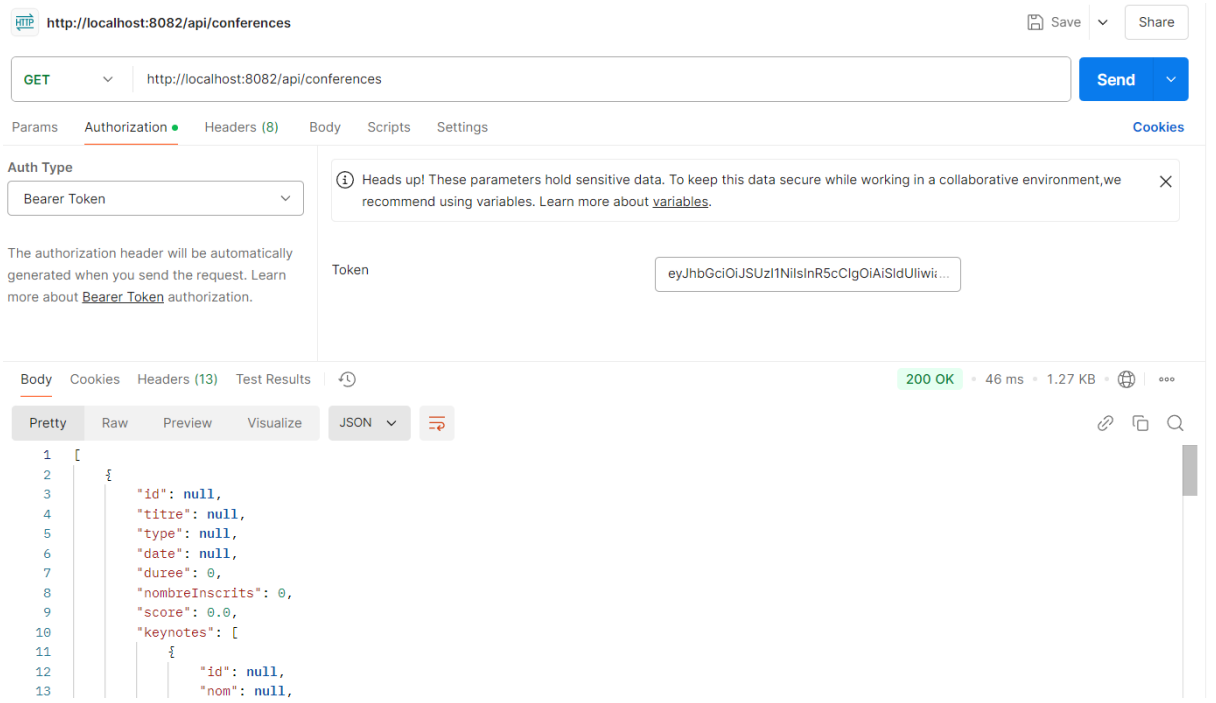
Modifier la config de properties

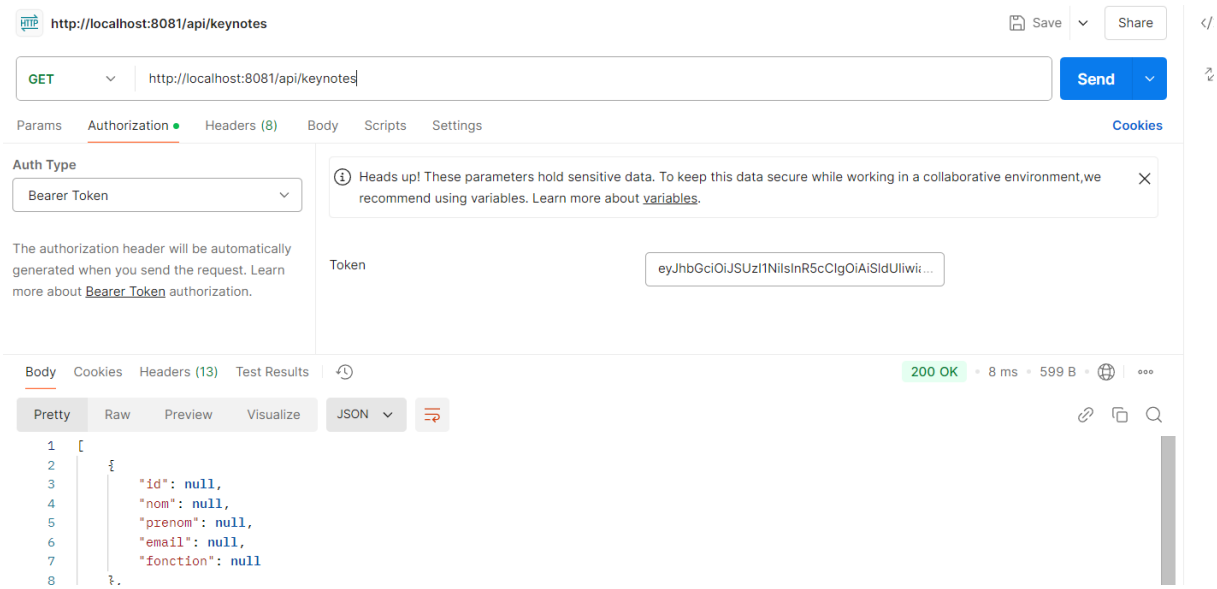
```properties
spring.application.name=keynote-service

server.port=8081
spring.datasource.url=jdbc:h2:mem:keynote-db
spring.h2.console.enabled=true

spring.cloud.config.enabled=false
spring.cloud.discovery.enabled=false

spring.security.oauth2.resourceserver.jwt.issuer-uri=http://localhost:8080/realms/exam-realm
spring.security.oauth2.resourceserver.jwt.jwk-set-uri=http://localhost:8080/realms/exam-realm/protocol/openid-connect/certs
```

```properties
pring.application.name=conference-service

erver.port=8082
pring.datasource.url=jdbc:h2:mem:conference-db
pring.h2.console.enabled=true

pring.cloud.config.enabled=false
pring.cloud.discovery.enabled=false

pring.security.oauth2.resourceserver.jwt.issuer-uri=http://localhost:8080/realms/exam-realm
pring.security.oauth2.resourceserver.jwt.jwk-set-uri=http://localhost:8080/realms/exam-realm/protocol/openid-connect/certs
```

Testing avec postman


POST
http://localhost:8080/realms/exam-realm/protocol/openid-connect/token
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Key
Value
Description
Bulk Edit
username
user2
password
1234
grant_type
password
client_id
exam-client-ang
Body
Cookies
Headers (11)
Test Results
200 OK
72 ms
2.88 KB
Pretty
Raw
Preview
Visualize
JSON
{
"access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJaX1VFV0ZPcjBNWV81RDhSZy1UTmhiblhXQmFpV1ljaF85N0JEWnd6dXNrIn0.
eyJleHAiOjE3MzQzNTE0NzgsImlhdCI6MTczNDM1MTE3OCwianRpIjoiNTYyMmIwNTYtMWNjNS00ZWUwLWE5NGMtY2QzNTdhOGI0ZDBkIiwiaXNzIjoiaHR0cDovL2xvY2Fs
aG9zdDo4MDgwL3JlYWxtcy9leGFtLXJlYWxtIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6IjEwMDM1MjIyLTJkM2UtNDRjNy1iZmE5LTliZWRmOWQwMTFjNiIsInR5cCI6IkJl
YXJlciIsImF6cCI6ImV4YW0tY2xpZW50LWFuZyIsInNlc3Npb25fc3RhdGUiOiJlYjNlZmZhNS0zMTZlLTRkMmMtYTEyYS04M2IzNTI2MzBhNDciLCJhY3IiOiIxIiwiYWxs
b3dlZC1vcmlnaW5zIjpbImh0dHA6Ly9sb2NhbGhvc3Q6NDIwMCJdLCJyZWFsbV9hY2Nlc3MiOnsicm9sZXMiOlsiZGVmYXVsdC1yb2xlcy1lY29tLXJlYWxtIiwib2ZmbGlu
ZV9hY2Nlc3MiLCJBRE1JTiIsInVtYV9hdXRob3JpemF0aW9uIiwiVVNFUiJdfSwicmVzb3VyY2VfYWNjZXNzIjp7ImFjY291bnQiOnsicm9sZXMiOlsibWFuYWdlLWFjY291
bnQiLCJtYW5hZ2UtYWNjb3VudC1saW5rcyIsInZpZXctcHJvZmlsZSJdfX0sInNjb3BlIjoicHJvZmlsZSBlbWFpbCIsInNpZCI6ImViM2VmZmE1LTMxNmUtNGQyYy1hMTJh
LTgzYjM1MjYzMGE0NyIsImVtYWlsX3ZlcmlmaWVkIjpmYWxzZSwibmFtZSI6IlVzZXIyRmlyc3ROYW1lIFVzZXIyTGFzdE5hbWUiLCJwcmVmZXJyZWRfdXNlcm5hbWUiOiJ1
c2VyMiIsImdpdmVuX25hbWUiOiJVc2VyMkZpcnN0TmFtZSIsImZhbWlseV9uYW1lIjoiVXNlcjJMYXN0TmFtZSIsImVtYWlsIjoidXNlcjJAZ21haWwuY29tIn0.
bxfmyKXxP6wcsMrK4oWuH97sMVJ3Ot9UuYrRIVNsOgBPB_VfxfGkyizRpwz4a1wbFcVdOG4Lg8UcjmkUN51XiFqj-lpnAIrPMDHrsjm03z5LVF4D1zfH5DMf3Na9hl34YTcB
wbHvdhq_NscRexnj-9WNtCTkBzOgmtgFU9HGKuQdDN32GQeMfAW401dKDAGw1wsrHVXerWY3qBCY7G7NYFWLsCOVM8VUqKWT7PJ6ItXtOiHbeD33sM0N3LgwpmX6LUhLEe_d
JZdHvo4elaPbIBF8QiZ9iFaHYkh_C3O2G7kjiI7Owwka21ygR26TOfSoLsy9YvM1WWWIq5q0SNV44w",
"expires_in": 300,
"refresh_expires_in": 1800,
"refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICIxOWVmZDg4MC05MDU0LTRhOTUtOWFkYS05MjFlOTZhMjJlMTEifQ.


http://localhost:8082/api/conferences
Save
Share
GET
http://localhost:8082/api/conferences
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
Auth Type
Bearer Token
The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.
Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment,we recommend using variables. Learn more about variables.
Token
eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwi...
Body
Cookies
Headers (13)
Test Results
200 OK
46 ms
1.27 KB
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": null,
"titre": null,
"type": null,
"date": null,
"duree": 0,
"nombreInscrits": 0,
"score": 0.0,
"keynotes": [
{
"id": null,
"nom": null,

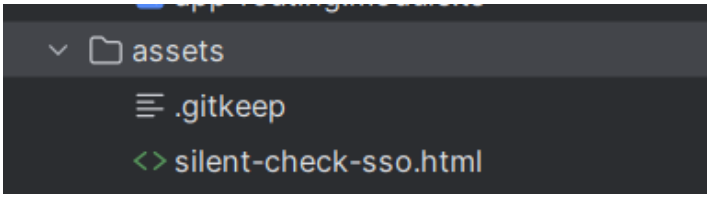


- Cote frontend :

```
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\angular-front-app> npm install keycloak-angular@15 keycloak-js

added 4 packages, and audited 904 packages in 5s

126 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\angular-front-app>
```

```
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
    KeycloakAngularModule
  ],
  providers: [
    {
      provide: APP_INITIALIZER,
      useFactory: initializeKeycloak,
      multi: true,
      deps: [KeycloakService]
    }
  ],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

```
1+ usages
function initializeKeycloak(keycloak: KeycloakService) {
  return () =>
    keycloak.init( options: {
      config: {
        url: 'http://localhost:8080',
        realm: 'ecom-realm',
        clientId: 'ecom-client-ang'
      },
      initOptions: {
        onLoad: 'check-sso',
        silentCheckSsoRedirectUri:
          window.location.origin + '/assets/silent-check-sso.html'
      }
    });
}
```




```
<html>
<body>
<script>
  parent.postMessage(location.href, location.origin);
</script>
</body>
</html>
```

Creation des guards :

```
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\angular-front-app> ng g g guards/auth
? Which type of guard would you like to create? CanActivate
CREATE src/app/guards/auth.guard.spec.ts (478 bytes)
CREATE src/app/guards/auth.guard.ts (133 bytes)
PS C:\Users\abdel\OneDrive\Desktop\GLSID_S5\examem_blanc\angular-front-app>
```

```
import { Injectable } from '@angular/core';
import {
  ActivatedRouteSnapshot,
  Router,
  RouterStateSnapshot
} from '@angular/router';
import { KeycloakAuthGuard, KeycloakService } from 'keycloak-angular';

no usages
@Injectable({
  providedIn: 'root'
})
export class AuthGuard extends KeycloakAuthGuard {
  no usages
  constructor(
    protected override readonly router: Router,
    protected readonly keycloak: KeycloakService
  ) {
    super(router, keycloak);
  }
```

```
no usages
public async isAccessAllowed(
  route: ActivatedRouteSnapshot,
  state: RouterStateSnapshot
) : Promise<boolean>  {
  // Force the user to log in if currently unauthenticated.
  if (!this.authenticated) {
    await this.keycloak.login( options: {
      redirectUri: window.location.origin + state.url
    });
  }

  // Get the roles required from the route.
  const requiredRoles = route.data['roles'];

  // Allow the user to proceed if no additional roles are required to access the route.
  if (!Array.isArray(requiredRoles) || requiredRoles.length === 0) {
    return true;
  }

  // Allow the user to proceed if all the required roles are present.
  return requiredRoles.every((role) => this.roles.includes(role));
 }
}
```

Modifie app-routing :

```
import {AuthGuard} from "./guards/auth.guard";

const routes: Routes = [
  {path : "keynotes", component : KeynotesComponent, canActivate : [AuthGuard], data : {roles :['ADMIN']}},
  {path : "conferences", component : ConferencesComponent, canActivate : [AuthGuard], data : {roles :['ADMIN']}},
];

1 usages
@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

8. Déployer l'application avec Docker et Docker compose