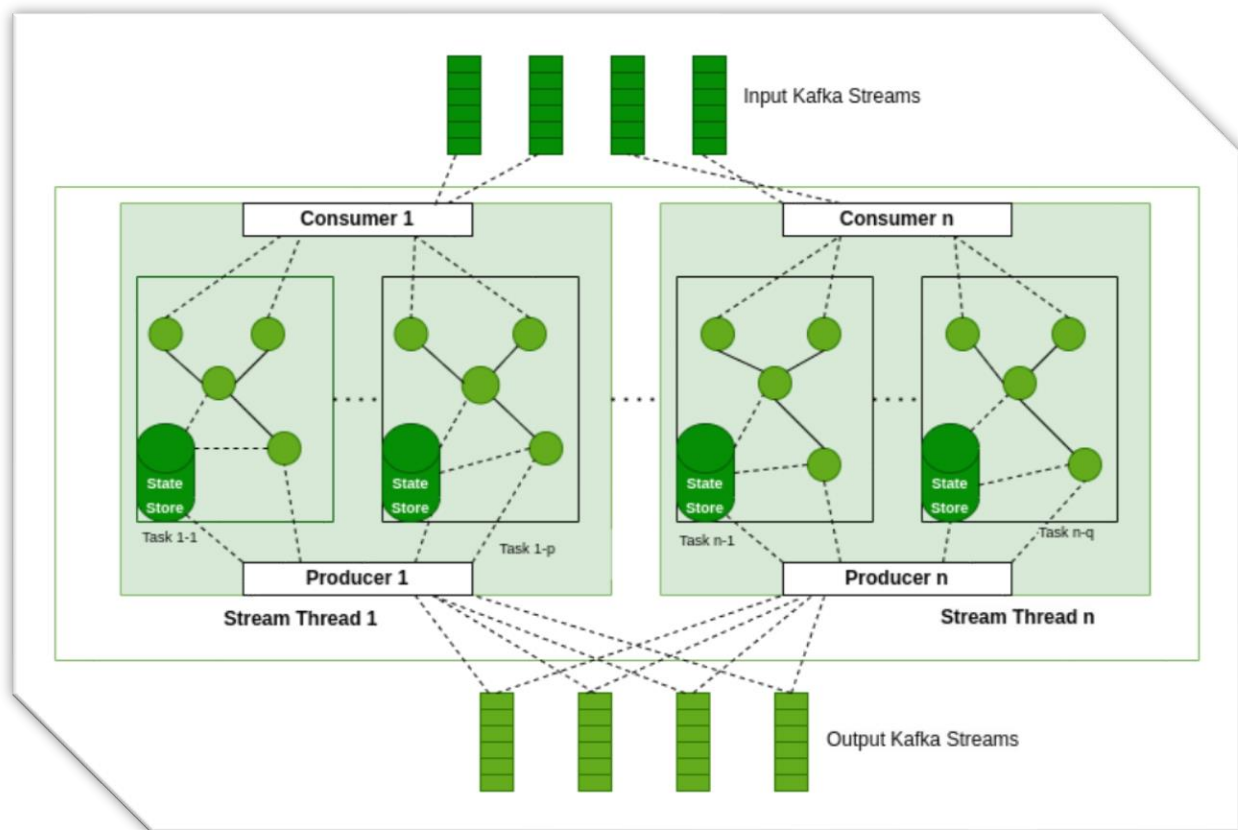


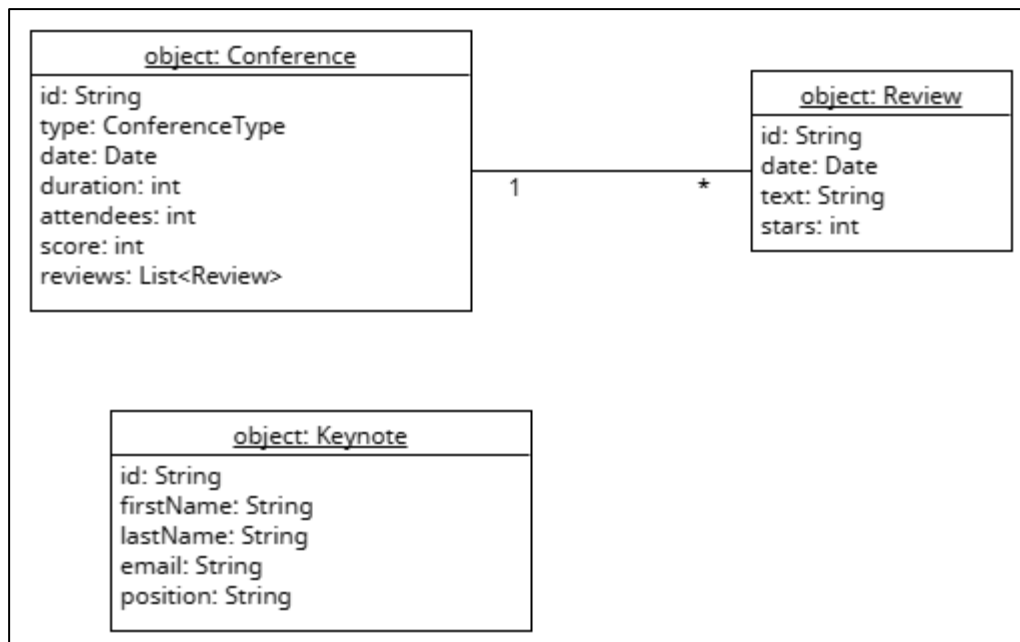
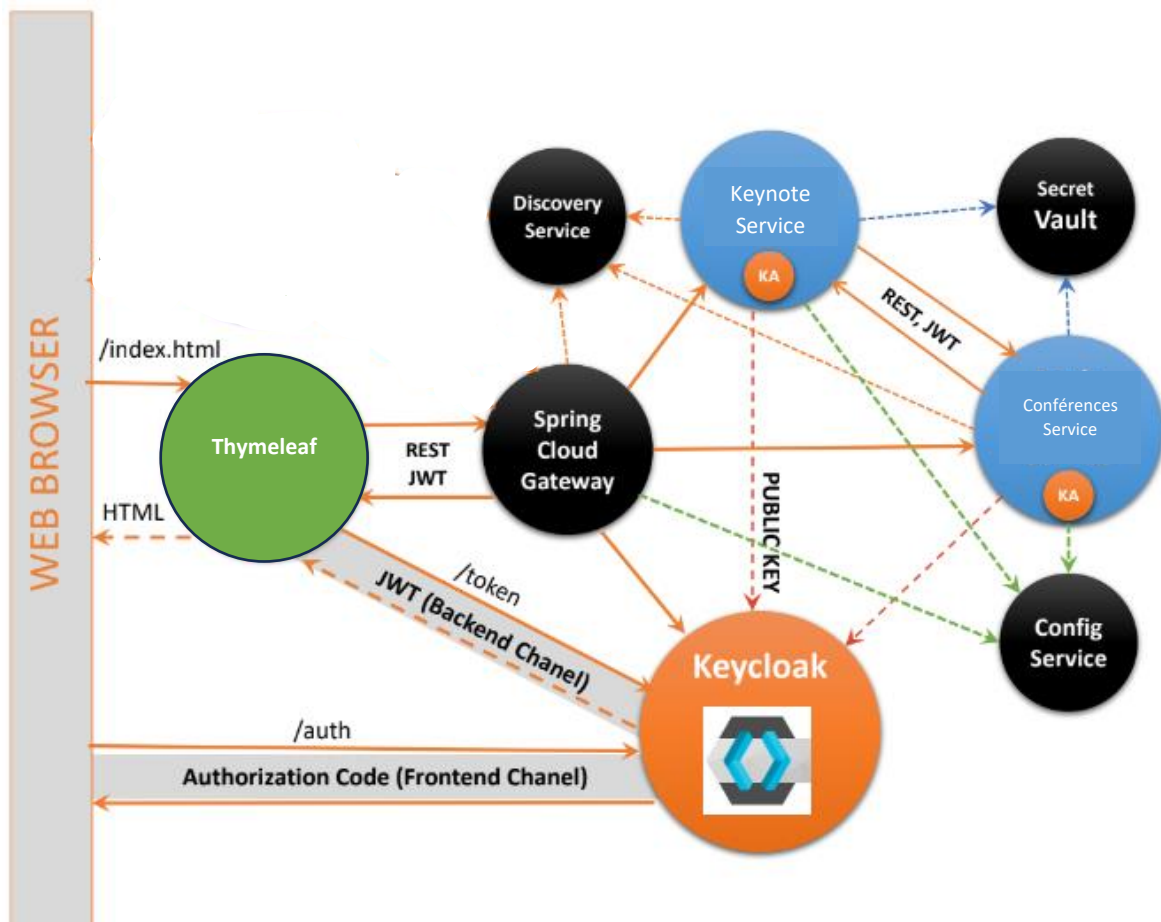
SOMMAIRE

Introduction.....	2
Travail à faire	3
A. Établir une architecture technique du projet.....	3
B. Créer un Projet Maven incluant les micro-services suivants : keynote-service, conference-service, gateway-service, discovery-service, config-service et angular-front-app.....	4
Conference-service.....	4
Keynote-service.....	5
gateway-service.....	7
discovery-service.....	7
config-service.....	7
C. Développer et tester les micro-services discovery-service et gateway-service et config-service	9
gateway-service.....	9
config-service.....	9
discovery-service.....	9
Teste	10
D. Développer et tester le micro-service Keynote-service (Entities, DAO, service, DTO, Mapper, RestController)	13
keynote-service.....	13
Teste	14
E. Développer et tester le micro-service conférence-service (Entities, DAO, service, DTO, Mapper, RestController, Client Rest Open Feign)	15
F. Développer un simple frontend web pour l'application	19
G. Sécuriser l'application avec une authentification Keycloak	25
H. Déployer l'application avec Docker et Docker compose.....	34
Conclusion	35



Le traitement parallèle occupe une place centrale dans le domaine du Big Data, où le volume, la variété et la vélocité des données posent des défis uniques. Face à cette réalité, les approches traditionnelles de traitement des données se révèlent souvent insuffisantes. C'est dans ce contexte que le traitement parallèle émerge comme une solution incontournable, permettant la distribution et l'exécution simultanée de tâches complexes sur des ressources informatiques multiples. Cette approche révolutionnaire transcende les limitations imposées par la capacité de traitement d'une seule machine, offrant une scalabilité et une efficacité sans précédent pour le traitement des ensembles de données massifs. Dans cette exploration, nous plongerons dans les fondements du traitement parallèle en Big Data, examinant les concepts clés, les modèles de programmation et les technologies qui propulsent cette révolution dans le domaine de l'informatique distribuée.

A. Établir une architecture technique du projet



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

Conference-service

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-oauth2-client</artifactId>
  </dependency>

  <dependency>
    <groupId>org.thymeleaf.extras</groupId>
    <artifactId>thymeleaf-extras-springsecurity6</artifactId>
    <version>3.1.0.M1</version>
  </dependency>

  <!-- https://mvnrepository.com/artifact/org.webjars/bootstrap -->
  <dependency>
    <groupId>org.webjars</groupId>
    <artifactId>bootstrap</artifactId>
    <version>5.3.2</version>
  </dependency>

  <dependency>
    <groupId>nz.net.ultraq.thymeleaf</groupId>
    <artifactId>thymeleaf-layout-dialect</artifactId>
    <version>3.2.1</version>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
  <!-- -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
```

```

</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-rest</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.cloud/spring-
cloud-starter-config -->
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>4.1.0</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.cloud/spring-
cloud-starter-consul-discovery -->
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-consul-discovery</artifactId>
    <version>4.1.0</version>
</dependency>

<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>

<dependency>
    <groupId>org.springdoc</groupId>
    <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
    <version>2.0.3</version>
</dependency>
</dependencies>

```

Keynote-service

```

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
<!--
    <dependency>-->
<!--
    <groupId>org.springframework.boot</groupId>-->
<!--
    <artifactId>spring-boot-starter-oauth2-resource-server</artifactId>--
>
<!--
    </dependency>-->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
</dependency>

```

```

<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>

<!-- -->

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-rest</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.cloud/spring-
cloud-starter-config -->
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-config</artifactId>
  <version>4.1.0</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.cloud/spring-
cloud-starter-consul-discovery -->
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-consul-discovery</artifactId>
  <version>4.1.0</version>
</dependency>

<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
  <scope>runtime</scope>
</dependency>
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>org.springdoc</groupId>
  <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
  <version>2.0.3</version>
</dependency>

</dependencies>

```

gateway-service

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-consul-discovery</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-gateway</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

discovery-service

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-consul-config</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-consul-discovery</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-vault-config</artifactId>
  </dependency>

  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

config-service

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
```



```
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-config-server</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-consul-discovery</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
</dependencies>
```

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

gateway-service

Application

```
@SpringBootApplication
public class GatewayServiceApplication {

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

    @Bean
    DiscoveryClientRouteDefinitionLocator dynamicRoutes(ReactiveDiscoveryClient rdc,
        DiscoveryLocatorProperties dlp){
        return new DiscoveryClientRouteDefinitionLocator(rdc, dlp);
    }

}
```

Application.properties

```
server.port=9999
spring.application.name=gateway-service
spring.config.import=optional:configserver:http://localhost:8888
```

config-service

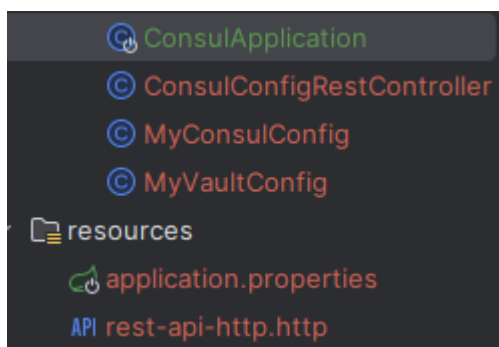
Application

```
@SpringBootApplication
@EnableConfigServer
@EnableDiscoveryClient
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

Application.properties

```
server.port=8888
spring.application.name=config-service
spring.cloud.config.server.git.uri=file:///C:/Tools/config-repo
```

discovery-service



consul-application

```
@Bean
CommandLineRunner commandLineRunner() {
    return args -> {
```

```
//Ajouter un secret <keypair>
vaultTemplate.opsForVersionedKeyValue("secret")
    .put("keypair",
        Map.of("privKey", "54321",
            "pubKey", "8999"));
};
};
```

Application.properties

```
server.port=8084
spring.application.name=config-consul

management.endpoints.web.exposure.include=*

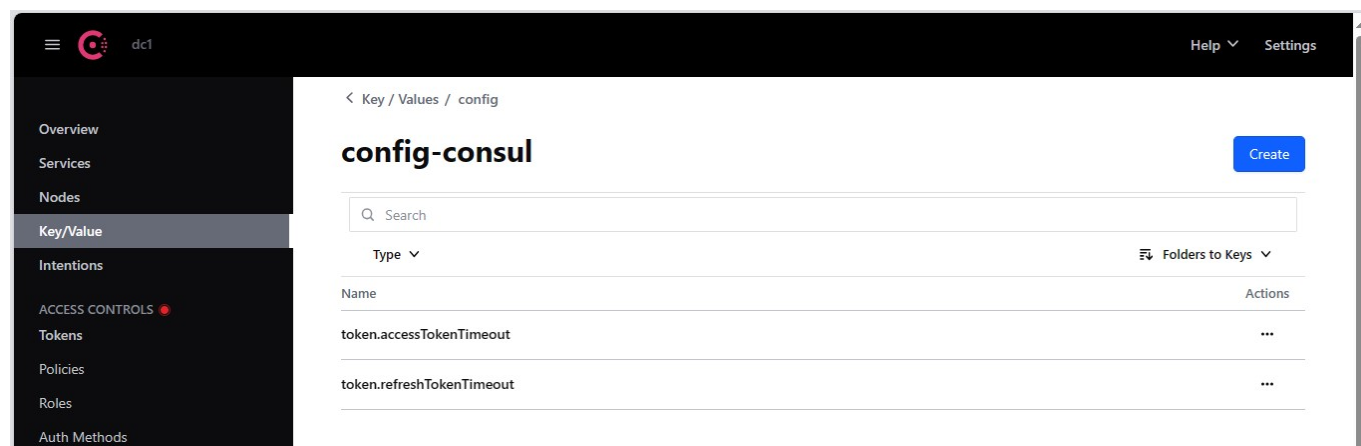
#by default
spring.config.import=optional:consul:, vault://

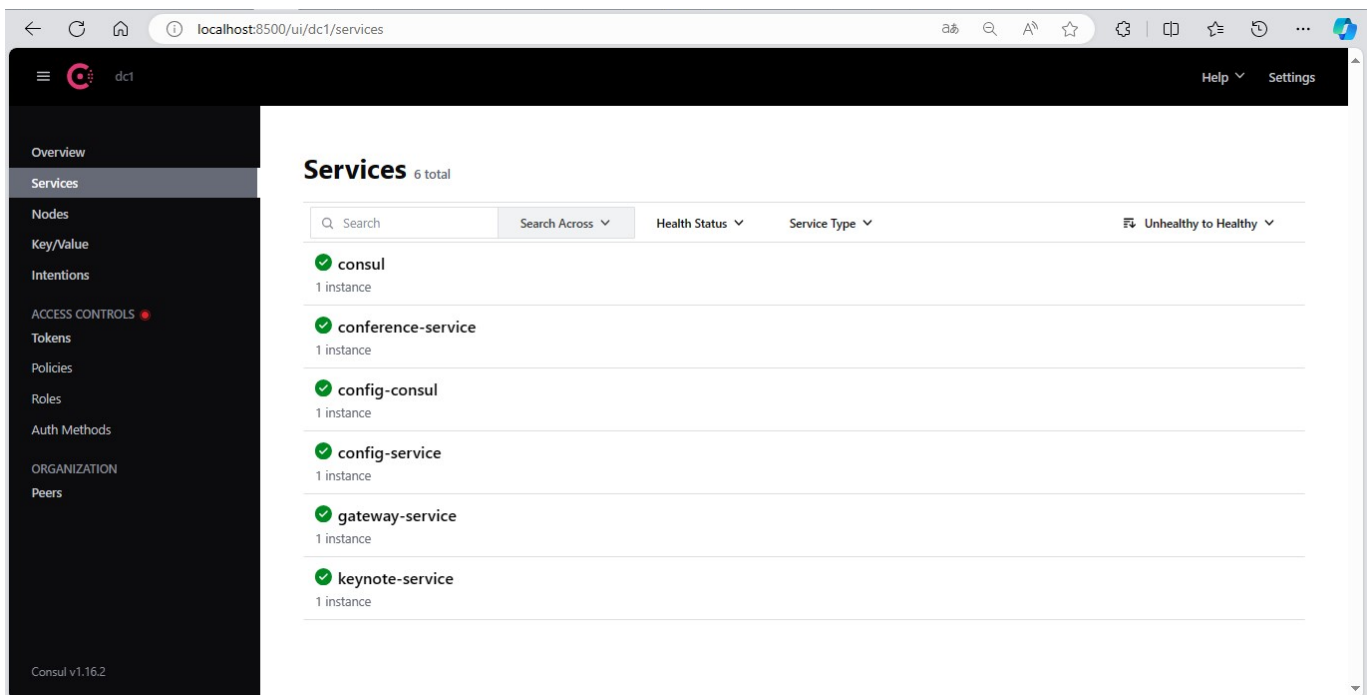
spring.cloud.vault.token=hvs.yn9wJcMXw3vDilCUCqtHaMTX
spring.cloud.vault.scheme=http
spring.cloud.vault.kv.enabled=true
```

Teste

Démarrer consul

```
C:\Windows\System32\cmd.exe - .\consul agent -server -bootstrap-expect=1 -data-dir=consul-data -ui-bind=192.168.0.109
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:41.959+0100 [WARN] agent: error getting server health from server: server=ACER error="context deadline
exceeded"
2023-12-25T08:31:42.304+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:42.382+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:42.947+0100 [WARN] agent: error getting server health from server: server=ACER error="rpc error gettin
g client: failed to get conn: dial tcp 192.168.0.109:0->192.168.0.109:8300: bind: The requested address is not valid in
its context."
2023-12-25T08:31:43.616+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:43.947+0100 [WARN] agent: error getting server health from server: server=ACER error="context deadline
exceeded"
2023-12-25T08:31:44.485+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:44.686+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:44.939+0100 [WARN] agent: error getting server health from server: server=ACER error="rpc error gettin
g client: failed to get conn: dial tcp 192.168.0.109:0->192.168.0.109:8300: bind: The requested address is not valid in
its context."
2023-12-25T08:31:45.404+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:45.435+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
2023-12-25T08:31:45.949+0100 [WARN] agent: error getting server health from server: server=ACER error="context deadline
exceeded"
2023-12-25T08:31:46.687+0100 [WARN] agent.http: This request used the token query parameter which is deprecated and wil
l be removed in Consul 1.17: logUrl="/v1/catalog/services?wait=2s&index=2129&token=<hidden>"
```





Démarrer vault

```
C:\Windows\System32\cmd.exe - vault server -dev

2023-12-24T12:49:44.867+0100 [INFO] rollback: Starting the rollback manager with 256 workers
2023-12-24T12:49:44.867+0100 [INFO] rollback: starting rollback manager
2023-12-24T12:49:44.867+0100 [INFO] core: restoring leases
2023-12-24T12:49:44.867+0100 [INFO] expiration: lease restore complete
2023-12-24T12:49:44.867+0100 [INFO] identity: entities restored
2023-12-24T12:49:44.867+0100 [INFO] identity: groups restored
2023-12-24T12:49:44.867+0100 [INFO] core: post-unseal setup complete
2023-12-24T12:49:44.867+0100 [INFO] core: vault is unsealed
2023-12-24T12:49:44.887+0100 [INFO] core: successful mount: namespace="" path=secret/ type=kv version=""
WARNING! dev mode is enabled! In this mode, Vault runs entirely in-memory
and starts unsealed with a single unseal key. The root token is already
authenticated to the CLI, so you can immediately begin using Vault.

You may need to set the following environment variables:

PowerShell:
    $env:VAULT_ADDR="http://127.0.0.1:8200"
cmd.exe:
    set VAULT_ADDR=http://127.0.0.1:8200

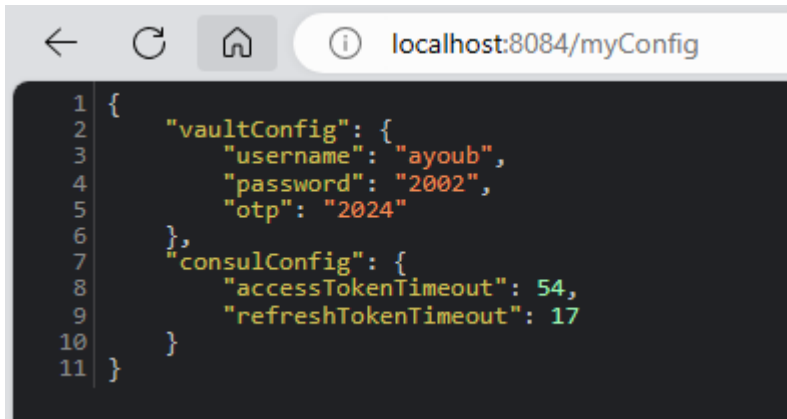
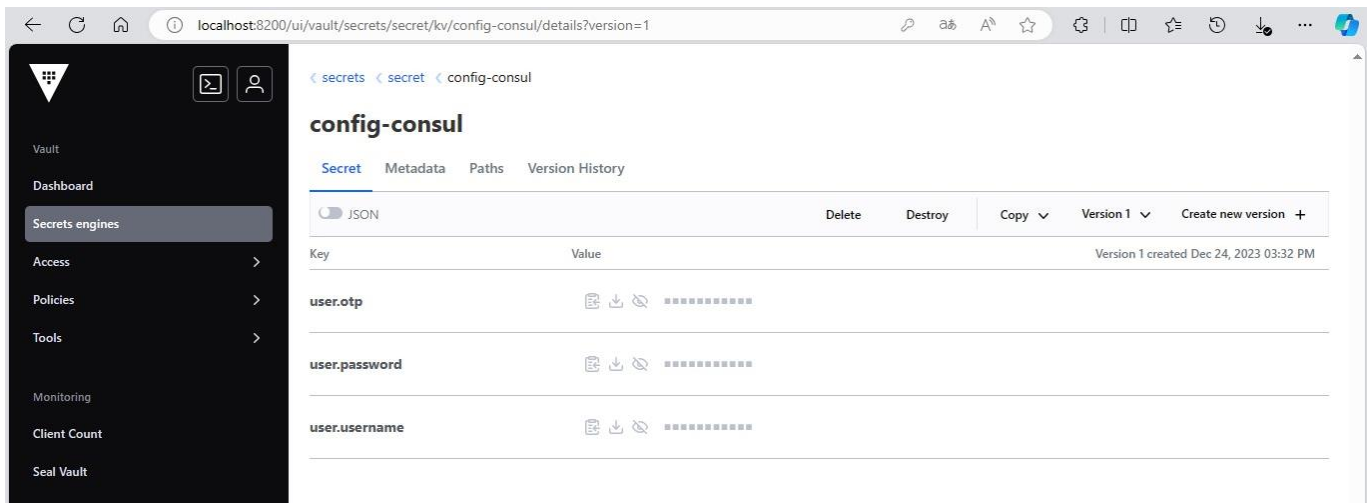
The unseal key and root token are displayed below in case you want to
seal/unseal the Vault or re-authenticate.

Unseal Key: W7nUmwwVT2BTdq2eaV4WEIZnqV81hi5PwB6L4KR+GSM=
Root Token: hvs.yn9wJcMXw3vDilCUCqtHaMTX

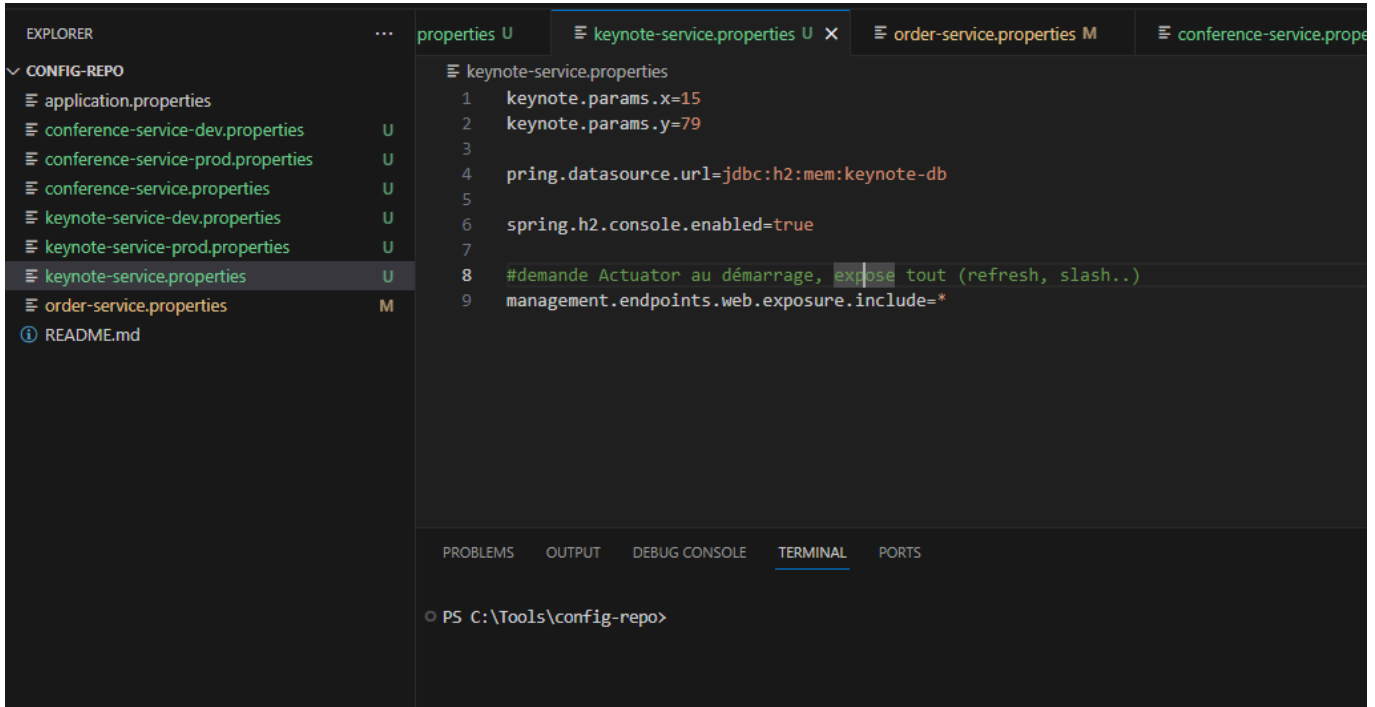
Development mode should NOT be used in production installations!
```

```
C:\Tools\vault>vault kv put secret/vaultConfig user.username=ayoub user.password=2002 user.otp=2024
===== Secret Path =====
secret/data/vaultConfig

===== Metadata =====
Key          Value
---          -
created_time  2023-12-24T14:26:51.8006953Z
custom_metadata <nil>
deletion_time n/a
destroyed     false
version       1
```



Config



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

keynote-service

entities

```
@Entity
@Data @NoArgsConstructor @AllArgsConstructor @Builder
public class Keynote {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String firstName;
    private String lastName;
    private String email;
    private String jobTitle;
}
```

Web RestController

```
@RestController
@RefreshScope
public class Controller {
    @Value("${global.params.p1}")
    private String p1;
    @Value("${global.params.p2}")
    private String p2;
    @Value("${inventory.params.x}")
    private String x;
    @Value("${inventory.params.y}")
    private String y;
    @GetMapping("/params")
    public Map<String, String> params() {
        return Map.of("p1", p1, "p2", p2, "x", x, "y", y);
    }
}
```

```
@RestController
public class KeynoteRestController {
    private KeynoteRepository keynoteRepository;

    public KeynoteRestController(KeynoteRepository keynoteRepository) {
        this.keynoteRepository = keynoteRepository;
    }
    @GetMapping("/keynotes")
    // @PreAuthorize("hasAuthority('ADMIN')")
    public List<Keynote> products() {
        return keynoteRepository.findAll();
    }
}
```

Application

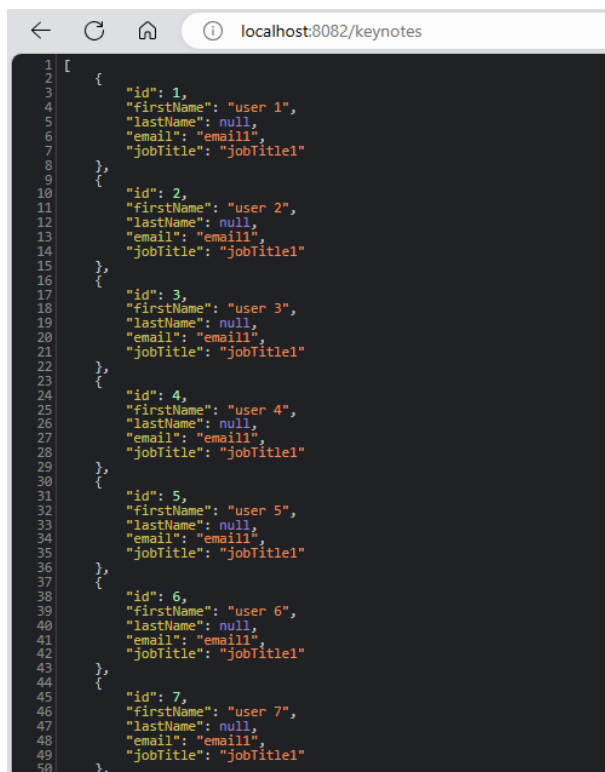
```
@Bean
CommandLineRunner start(KeynoteRepository keynoteRepository) {
    return args -> {
        Random random = new Random();
        for (int i = 1; i < 10; i++) {
            keynoteRepository.saveAll(List.of(
                Keynote.builder()
                    .firstName("user " + i)
                    .email("email" + i)
                    .jobTitle("jobTitle" + i)
                    .build()
            ));
        }
    }
}
```

```
};  
}
```

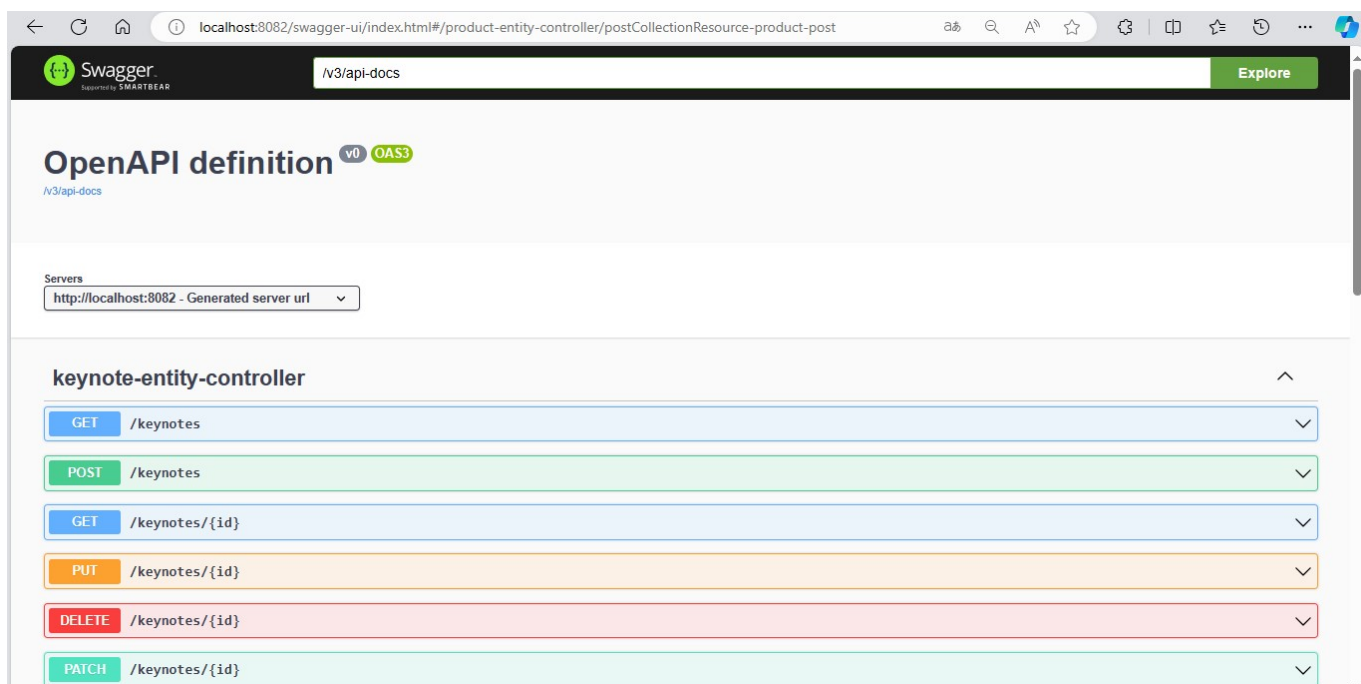
application.properties

```
server.port=8082  
spring.application.name=keynote-service  
spring.config.import=optional:configserver:http://localhost:8888  
  
#demande Actuator au démarrage, expose tout (refresh, slash..)  
management.endpoints.web.exposure.include=*
```

Teste



```
1 [
2   {
3     "id": 1,
4     "firstName": "user 1",
5     "lastName": null,
6     "email": "email1",
7     "jobTitle": "jobTitle1"
8   },
9   {
10    "id": 2,
11    "firstName": "user 2",
12    "lastName": null,
13    "email": "email1",
14    "jobTitle": "jobTitle1"
15  },
16  {
17    "id": 3,
18    "firstName": "user 3",
19    "lastName": null,
20    "email": "email1",
21    "jobTitle": "jobTitle1"
22  },
23  {
24    "id": 4,
25    "firstName": "user 4",
26    "lastName": null,
27    "email": "email1",
28    "jobTitle": "jobTitle1"
29  },
30  {
31    "id": 5,
32    "firstName": "user 5",
33    "lastName": null,
34    "email": "email1",
35    "jobTitle": "jobTitle1"
36  },
37  {
38    "id": 6,
39    "firstName": "user 6",
40    "lastName": null,
41    "email": "email1",
42    "jobTitle": "jobTitle1"
43  },
44  {
45    "id": 7,
46    "firstName": "user 7",
47    "lastName": null,
48    "email": "email1",
49    "jobTitle": "jobTitle1"
50  },
51 ]
```



Swagger
v3/api-docs

OpenAPI definition v0 OAS3

Servers
http://localhost:8082 - Generated server url

keynote-entity-controller

- GET /keynotes
- POST /keynotes
- GET /keynotes/{id}
- PUT /keynotes/{id}
- DELETE /keynotes/{id}
- PATCH /keynotes/{id}

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

entities

```
@Entity
@NoArgsConstructor @AllArgsConstructor @Getter @Setter @ToString @Builder
public class Conference {
    private Long id;
    private String titre;
    private ConferenceType type; // académique ou commerciale
    private Date date;
    private int duree; // en minutes
    private int nombreInscrits;
    private int score;

    // Relation avec les reviews
    @OneToMany(mappedBy = "conference", cascade = CascadeType.ALL)
    private List<Review> reviews;
}
```

Enum

```
public enum ConferenceType {
    ACADEMIC,
    COMMERCIAL
}
```

model

```
@NoArgsConstructor @AllArgsConstructor @Getter @Setter @Builder
public class Keynote {
    private Long id;
    private String firstName;
    private String lastName;
    private String email;
    private String jobTitle;
}
```

Web RestController

```
@RestController
@RefreshScope
public class Controller {
    @Value("${global.params.p1}")
    private String p1;
    @Value("${global.params.p2}")
    private String p2;
    @Value("${conference.params.x}")
    private String x;
    @Value("${conference.params.y}")
    private String y;
    @GetMapping("/params")
    public Map<String, String> params() {
        return Map.of("p1", p1, "p2", p2, "x", x, "y", y);
    }
}
```

```
@Controller
public class ConferenceController {
    private ConferenceRepository conferenceRepository;
    private ClientRegistrationRepository clientRegistrationRepository;

    public ConferenceController(ConferenceRepository conferenceRepository,
    ClientRegistrationRepository clientRegistrationRepository) {
        this.conferenceRepository = conferenceRepository;
        this.clientRegistrationRepository = clientRegistrationRepository;
    }

    @GetMapping("/index")
```



```

@PreAuthorize("hasAuthority('USER')")
public String index() {
    return "index";
}

@GetMapping("/")
public String index1() {
    return "index";
}

@GetMapping("/conferences")
@PreAuthorize("hasAuthority('ADMIN')")
public String conferences(Model model) {
    List<Conference> conferenceList = conferenceRepository.findAll();
    model.addAttribute("conferences", conferenceList);
    return "conferences";
}

@GetMapping("/keynotes")
public String keynotes(Model model) {
    SecurityContext context = SecurityContextHolder.getContext();
    Authentication authentication = context.getAuthentication();
    OAuth2AuthenticationToken oAuth2AuthenticationToken =
(OAuth2AuthenticationToken) authentication;
    DefaultOidcUser oidcUser = (DefaultOidcUser)
oAuth2AuthenticationToken.getPrincipal();
    String jwtTokenValue=oidcUser.getIdToken().getTokenValue();
    RestClient restClient = RestClient.create("http://localhost:8082");
    List<Keynote> keynotes = restClient.get()
        .uri("/keynotes")
        .headers(httpHeaders -> httpHeaders.set(HttpHeaders.AUTHORIZATION,
"Bearer "+jwtTokenValue))
        .retrieve()
        .body(new ParameterizedTypeReference<>() {});
    model.addAttribute("keynotes", keynotes);
    return "keynotes";
}

@GetMapping("/auth")
@ResponseBody
public Authentication authentication(Authentication authentication) {
    return authentication;
}

@GetMapping("/notAuthorized")
public String notAuthorized() {
    return "notAuthorized";
}

@GetMapping("/oauth2Login")
public String oauth2Login(Model model) {
    String authorizationRequestBaseUrl = "oauth2/authorization";
    Map<String, String> oauth2AuthenticationUrls = new HashMap();
    Iterable<ClientRegistration> clientRegistrations =
(Iterable<ClientRegistration>) clientRegistrationRepository;
    clientRegistrations.forEach(registration ->{
        oauth2AuthenticationUrls.put(registration.getClientName(),
            authorizationRequestBaseUrl + "/" +
registration.getRegistrationId());
    });
    model.addAttribute("urls", oauth2AuthenticationUrls);
    return "oauth2Login";
}
}

```

Application

```

@SpringBootApplication
public class ConferenceFrontThymeleafAppApplication {

```

```

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

@Bean
CommandLineRunner commandLineRunner(ConferenceRepository conferenceRepository) {
    return args -> {
        conferenceRepository.save(Conference.builder().titre("conference
1").type(ConferenceType.ACADEMIC).duree(60).build());
        conferenceRepository.save(Conference.builder().titre("conference
2").type(ConferenceType.ACADEMIC).duree(120).build());
        conferenceRepository.save(Conference.builder().titre("conference
3").type(ConferenceType.COMMERCIAL).duree(60).build());
        conferenceRepository.save(Conference.builder().titre("conference
4").type(ConferenceType.COMMERCIAL).duree(30).build());

    };
}
}

```

application.properties

```

spring.datasource.url=jdbc:h2:mem:Conference-db
spring.h2.console.enabled=true

spring.security.oauth2.client.registration.google.clientId=260777653322-
4oagm6rn1qtnrm0ubp98o2jia5cnbb89.apps.googleusercontent.com
spring.security.oauth2.client.registration.google.client-secret=GOCSPX-
xmR3D5IuK8J0kNymjjMQof8UUkUJ
spring.security.oauth2.client.provider.google.user-name-attribute=email

spring.security.oauth2.client.registration.github.clientId=edb49a0a8c36f1e84464
spring.security.oauth2.client.registration.github.client-
secret=2bb2ad89fb0440cfb5257aa9740ac7fe8f6be4af
spring.security.oauth2.client.provider.github.user-name-attribute=login

spring.security.oauth2.client.registration.keycloak.client-name=keycloak
spring.security.oauth2.client.registration.keycloak.client-id=enset-Conference-client
spring.security.oauth2.client.registration.keycloak.client-
secret=sRrSOVPk2EuOdeMzkFTZhuoCv36ArVwk
spring.security.oauth2.client.registration.keycloak.scope=openid,profile,email,offline_
e_access
spring.security.oauth2.client.registration.keycloak.authorization-grant-
type=authorization_code
spring.security.oauth2.client.registration.keycloak.redirect-
uri=http://localhost:8081/login/oauth2/code/enset-Conference-client
spring.security.oauth2.client.provider.keycloak.issuer-
uri=http://localhost:8080/realms/enset-realm

spring.security.oauth2.client.provider.keycloak.user-name-
attribute=preferred_username
#spring.security.oauth2.resourceserver.jwt.issuer-
uri=http://localhost:8080/realms/ebank-realm

###

server.port=8081
spring.application.name=conference-service
spring.config.import=optional:configserver:http://localhost:8888

#demande Actuator au démarrage, expose tout (refresh, slash..)
management.endpoints.web.exposure.include=*

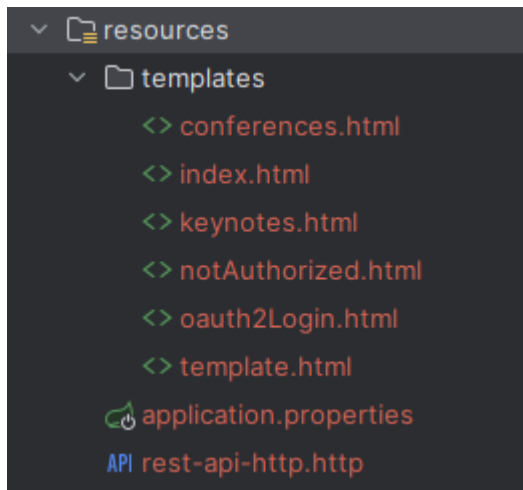
```

Test

The screenshot displays the Swagger UI interface in a web browser. The address bar shows the URL: `localhost:8081/swagger-ui/index.html#/customer-entity-controller/postCollectionResource-customer-post`. The Swagger logo and version `v0` are visible in the top left. The main heading is **OpenAPI definition** with a sub-label `v0` and a green **OAS3** badge. Below this, a **Servers** dropdown menu is set to `http://localhost:8081 - Generated server url`. The central section, titled **conference-entity-controller**, lists five API endpoints with their respective HTTP methods and expandable details (indicated by chevrons):

- GET** `/conferences`
- POST** `/conferences`
- GET** `/conferences/{id}`
- PUT** `/conferences/{id}`
- DELETE** `/conferences/{id}`

F. Développer un simple frontend web pour l'application



Conferences

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
      layout:decorate="template">
</html>
<head>
  <meta charset="UTF-8">
  <title>Conference</title>
  <link rel="stylesheet" href="/webjars/bootstrap/5.3.2/css/bootstrap.min.css">
</head>
<body>
  <div layout:fragment="content">
    <div class="ps-3">
      <div class="card">
        <div class="card-body">
          <table class="table">
            <thead>
              <tr>
                <th>ID</th><th>Titre</th><th>Type</th><th>Date</th><th>Durée</th>
              </tr>
            </thead>
            <tbody>
              <tr th:each="c:${conferences}">
                <td th:text="${c.id}"></td>
                <td th:text="${c.titre}"></td>
                <td th:text="${c.type}"></td>
                <td th:text="${c.date}"></td>
                <td th:text="${c.duree}"></td>
              </tr>
            </tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</body>
</html>
```

Keynotes

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
      layout:decorate="template">
</html>
<head>
```

```

<meta charset="UTF-8">
<title>Keynote</title>
<link rel="stylesheet" href="/webjars/bootstrap/5.3.2/css/bootstrap.min.css">
</head>
<body>
  <div layout:fragment="content">
    <div class="ps-3">
      <div class="card">
        <div class="card-body">
          <table class="table">
            <thead>
              <tr>
                <th>ID</th><th>First Name</th><th>Email</th><th>Job
Title</th>
            </tr>
            </thead>
            <tbody>
              <tr th:each="p:${keynotes}">
                <td th:text="{p.id}"></td>
                <td th:text="{p.firstName}"></td>
                <td th:text="{p.email}"></td>
                <td th:text="{p.jobTitle}"></td>
              </tr>
            </tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</body>
</html>

```

Index

```

<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
  layout:decorate="template">
</html>
<head>
  <meta charset="UTF-8">
  <title>Customers</title>
  <link rel="stylesheet" href="/webjars/bootstrap/5.3.2/css/bootstrap.min.css">
</head>
<body>
  <div layout:fragment="content">
    <div class="ps-3">
      <div class="card">
        <div class="card-body">
          <div>
            <h1>Welcome to the dashboard </h1>
            <p>....</p>
          </div>
        </div>
      </div>
    </div>
  </div>
</body>
</html>

```

Template

```

<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
</html>
<head>
  <meta charset="UTF-8">
  <title>Template Conference</title>

```

```

<link rel="stylesheet" type="text/css"
href="/webjars/bootstrap/5.3.2/css/bootstrap.min.css">
<script src="webjars/bootstrap/5.3.2/js/bootstrap.bundle.js"></script>
</head>
<body>
<nav class="navbar navbar-expand-lg bg-dark navbar-dark">
  <div class="container-fluid">
    <a class="navbar-brand" href="https://ayoub-etoullali.netlify.app/">
      
</a>
<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
  <span class="navbar-toggler-icon"></span>
</button>
<div style="color: white" class="collapse navbar-collapse"
id="navbarSupportedContent">
  <ul class="navbar-nav me-auto mb-2 mb-lg-0">
    <li class="nav-item">
      <a class="nav-link active" aria-current="page" th:href="@{/index}">Home</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" th:href="@{/conferences}">Conferences</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" th:href="@{/keynotes}">Keynotes</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#" tabindex="-1" aria-
disabled="true">Disabled</a>
    </li>
  </ul>
  <ul class="navbar">
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-bs-toggle="dropdown" aria-expanded="false">
        <span th:text="{#authentication.name}"></span>
      </a>
      <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
        <li><a class="dropdown-item" th:href="@{/login}">Login</a></li>
        <li>
          <form th:action="@{/logout}" method="post">

```



```

        <button class="btn btn-success">Logout</button>
    </form>
</li>
<li><hr class="dropdown-divider"></li>
<li><a class="dropdown-item" href="#">Profile</a></li>
</ul>
</li>
</ul>
</div>
</div>
</nav>
<div layout:fragment="content">

</div>
</body>
</html>

```

Oauth2login

```

<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
    xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
    layout:decorate="template"
>
<head>
    <meta charset="UTF-8">
    <title>Customers</title>
    <link rel="stylesheet" href="/webjars/bootstrap/5.3.2/css/bootstrap.min.css">
</head>
<body>
    <div layout:fragment="content">
        <div class="ps-3">
            <div class="card">
                <div class="card-body">
                    <ul class="list-group">
                        <li class="list-group-item" th:each="url:${urls}">
                            <div class="d-grid">
                                <a class="btn btn-outline-info" th:text="${url.key}"
th:href="${url.value}"></a>
                            </div>
                        </li>
                    </ul>
                </div>
            </div>
        </div>
    </div>
</body>
</html>

```

notAuthorized

```

<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
    xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
    layout:decorate="template"
>
<head>
    <meta charset="UTF-8">
    <title>Customers</title>
    <link rel="stylesheet" href="/webjars/bootstrap/5.3.2/css/bootstrap.min.css">
</head>
<body>
    <div layout:fragment="content">
        <div class="ps-3">
            <div class="card">
                <div class="card-body">
                    <h2 class="text-danger">Not authorized</h2>
                </div>
            </div>
        </div>
    </div>
</body>
</html>

```



```
    </div>
  </div>
</body>
</html>
```

G. Sécuriser l'application avec une authentification Keycloak

Coference

```
package ma.enset.conferencefrontthymeleafapp.sec;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.Customizer;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.authority.mapping.GrantedAuthoritiesMapper;
import org.springframework.security.oauth2.client.oidc.web.logout.OidcClientInitiatedLogoutSuccessHandler;
import org.springframework.security.oauth2.client.registration.ClientRegistrationRepository;
import org.springframework.security.oauth2.core.oidc.user.OidcUserAuthority;
import org.springframework.security.oauth2.core.user.OAuth2UserAuthority;
import org.springframework.security.web.SecurityFilterChain;

import java.util.*;

@Configuration
@EnableWebSecurity
@EnableMethodSecurity(prePostEnabled = true)
public class SecurityConfig {
    private ClientRegistrationRepository clientRegistrationRepository;

    public SecurityConfig(ClientRegistrationRepository clientRegistrationRepository) {
        this.clientRegistrationRepository = clientRegistrationRepository;
    }

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        return http
            .csrf(Customizer.withDefaults())
            .authorizeHttpRequests(ar->ar.requestMatchers("/", "/oauth2Login/**", "/webjars/**", "/h2-console/**").permitAll())
            .authorizeHttpRequests(ar->ar.anyRequest().authenticated())
            .headers(h->h.frameOptions(fo->fo.disable()))
            .csrf(csrf->csrf.ignoringRequestMatchers("/h2-console/**"))
            .oauth2Login(al->
                al.loginPage("/oauth2Login")
                .defaultSuccessUrl("/")
            )
            .logout((logout) -> logout
                .logoutSuccessHandler(oidcLogoutSuccessHandler())
                .logoutSuccessUrl("/")
                .permitAll()
                .clearAuthentication(true)
                .deleteCookies("JSESSIONID")
                .exceptionHandling(eh->eh.accessDeniedPage("/notAuthorized"))
            )
            .build();

        private OidcClientInitiatedLogoutSuccessHandler oidcLogoutSuccessHandler() {
            final OidcClientInitiatedLogoutSuccessHandler oidcLogoutSuccessHandler =
                new
```

```

OidcClientInitiatedLogoutSuccessHandler(this.clientRegistrationRepository);

oidcLogoutSuccessHandler.setPostLogoutRedirectUri("{baseUrl}?logoutsuccess=true");
return oidcLogoutSuccessHandler;
}

@Bean
public GrantedAuthoritiesMapper userAuthoritiesMapper() {
    return (authorities) -> {
        final Set<GrantedAuthority> mappedAuthorities = new HashSet<>();
        authorities.forEach((authority) -> {
            if (authority instanceof OidcUserAuthority oidcAuth) {
mappedAuthorities.addAll(mapAuthorities(oidcAuth.getIdToken().getClaims()));
                System.out.println(oidcAuth.getAttributes());
            } else if (authority instanceof OAuth2UserAuthority oauth2Auth) {
mappedAuthorities.addAll(mapAuthorities(oauth2Auth.getAttributes()));
            }
        });
        return mappedAuthorities;
    };
}

private List<SimpleGrantedAuthority> mapAuthorities(final Map<String, Object>
attributes) {
    final Map<String, Object> realmAccess = ((Map<String,
Object>)attributes).getOrDefault("realm_access", Collections.emptyMap());
    final Collection<String> roles =
((Collection<String>)realmAccess.getOrDefault("roles", Collections.emptyList()));
    return roles.stream()
        .map((role) -> new SimpleGrantedAuthority(role))
        .toList();
}
}

```

keynote

JwtAuthConverter

```

@Component
public class JwtAuthConverter implements Converter<Jwt, AbstractAuthenticationToken>
{
    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"));
    }

    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());
    }
}

```

SecurityConfig

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

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

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws
Exception {
        return http
            .authorizeHttpRequests(ar->ar.anyRequest().authenticated())
            .oauth2ResourceServer(o2->o2.jwt(jwt-
>jwt.jwtAuthenticationConverter(jwtAuthConverter)))
            .headers(h->h.frameOptions(fo->fo.disable()))
            .csrf(csrf->csrf.ignoringRequestMatchers("/h2-console/**"))
            .build();
    }
}
```

Interfaces

Conference : USER, ADMIN

Keynote : ADMIN

Google

Google Cloud console interface showing the 'Identifiants' (Credentials) page. The page displays three sections: 'Clés API' (API Keys), 'ID clients OAuth 2.0' (OAuth 2.0 Client IDs), and 'Comptes de service' (Service Accounts).

Clés API

Nom	Date de création	Restrictions	Actions
Aucune clé API à afficher			

ID clients OAuth 2.0

Nom	Date de création	Type	ID client	Actions
ENSET App	5 déc. 2023	Application Web	260777653322-4oag...	[Edit] [Delete] [Download]

Comptes de service

E-mail	Nom	Actions
Aucun compte de service à afficher		

← ↻ 🏠 ⓘ localhost:8080/admin/master/console/#/enset-realm/users

🔍 🗑️ 🌟 🔄 📄 ⌵ ⌚ ⋮ 👤

☰ 🔑 KEYCLOAK

⌚ admin 👤

enset-realm ▾

Manage

Clients

Client scopes

Realm roles

Users

Groups

Sessions

Events

Configure

Realm settings

Authentication

Identity providers

User federation

Users

Users are the users in the current realm. [Learn more](#)

User list

Default search ▾ 🔍 Search user ➔

Add user

Delete user

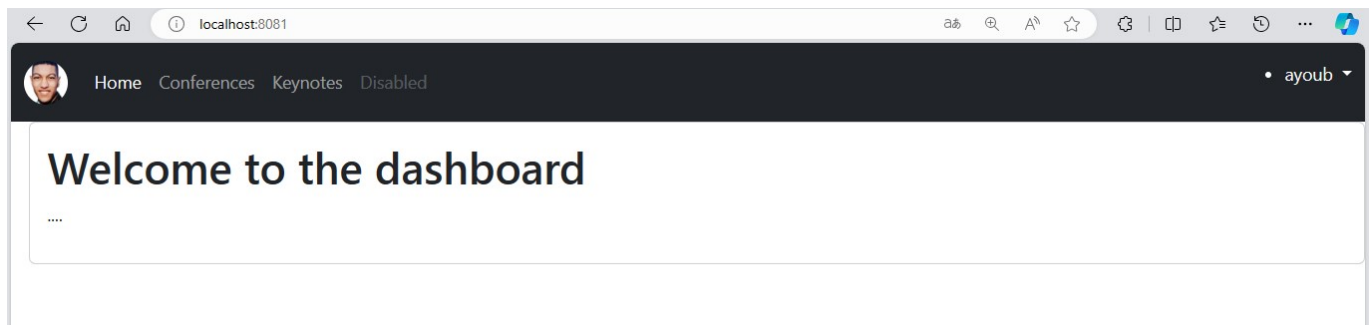
1-6 < >

<input type="checkbox"/>	Username	Email	Last name	First name	Status	
<input type="checkbox"/>	ayoub	🚫 —	—	—	—	⋮
<input type="checkbox"/>	hayat	🚫 —	—	—	—	⋮
<input type="checkbox"/>	ihssan	🚫 ihssan@gmail.com	Ftah	ihssan	—	⋮
<input type="checkbox"/>	karima	🚫 —	—	—	—	⋮
<input type="checkbox"/>	radouan	🚫 —	—	—	—	⋮
<input type="checkbox"/>	samira	🚫 —	—	—	—	⋮

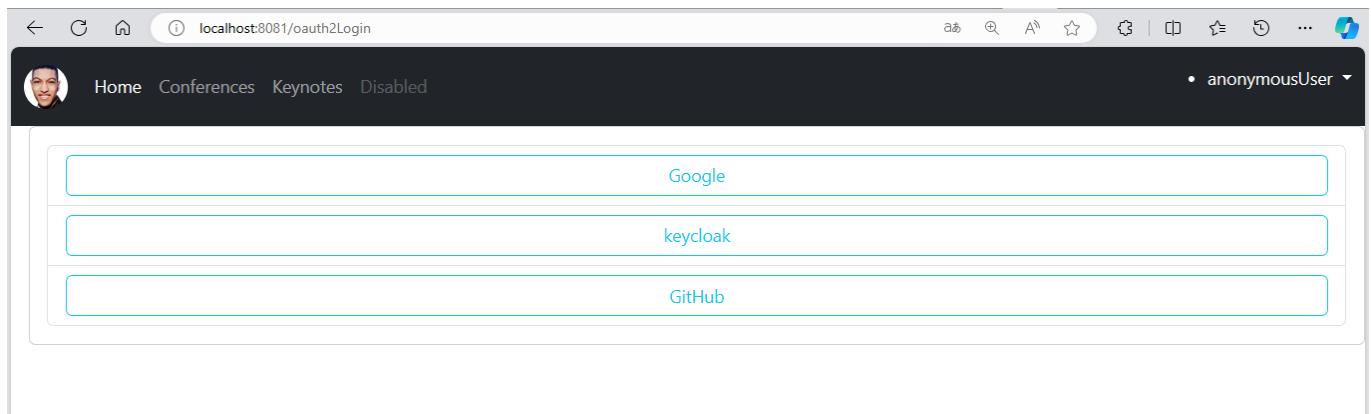
1-6 < >

[illegible]

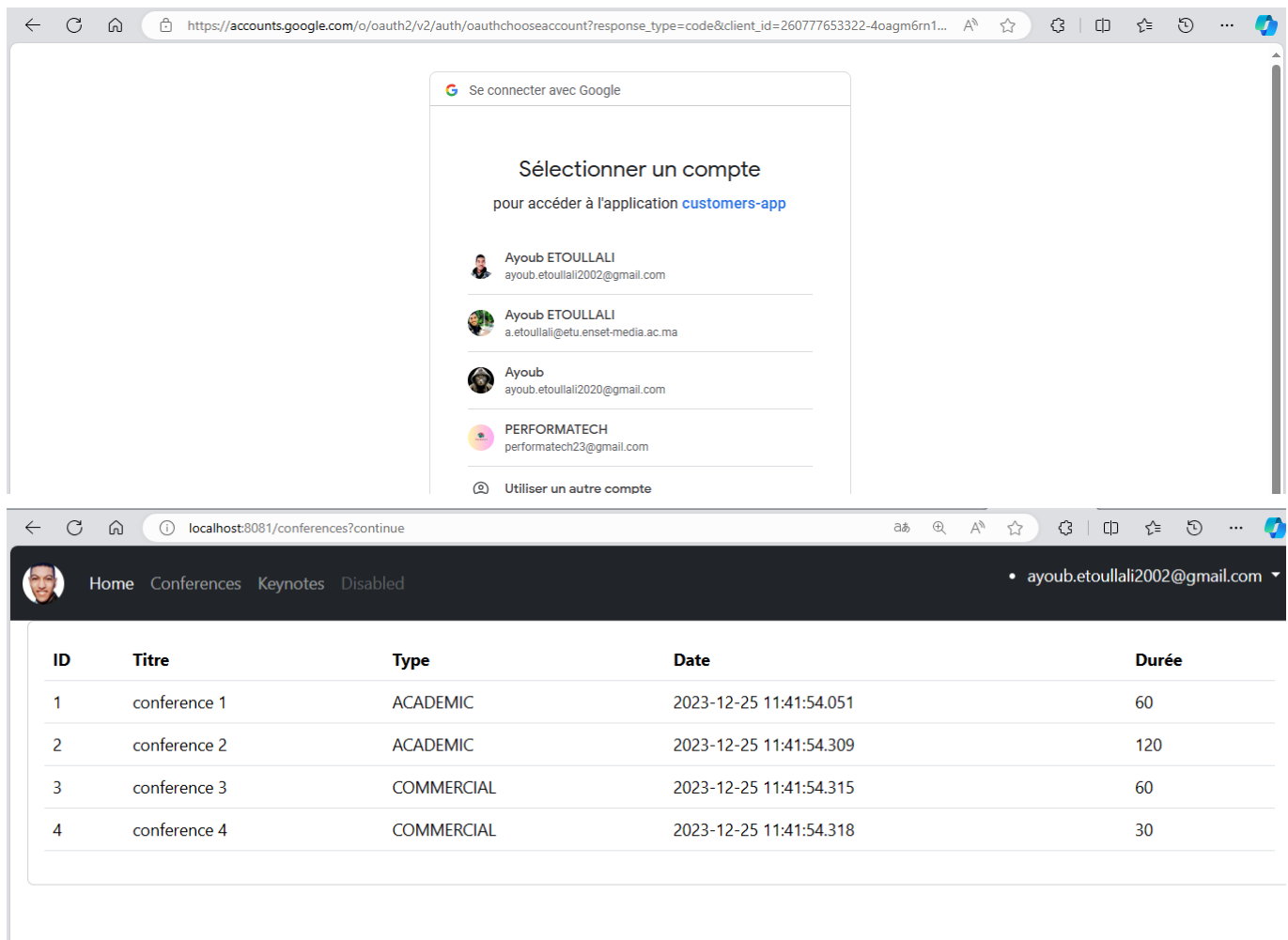
[illegible]

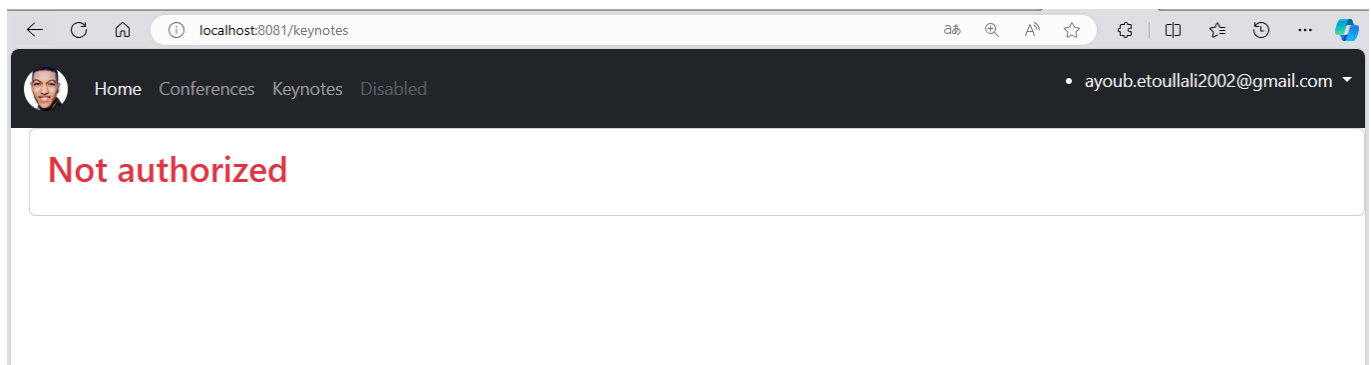


authentification

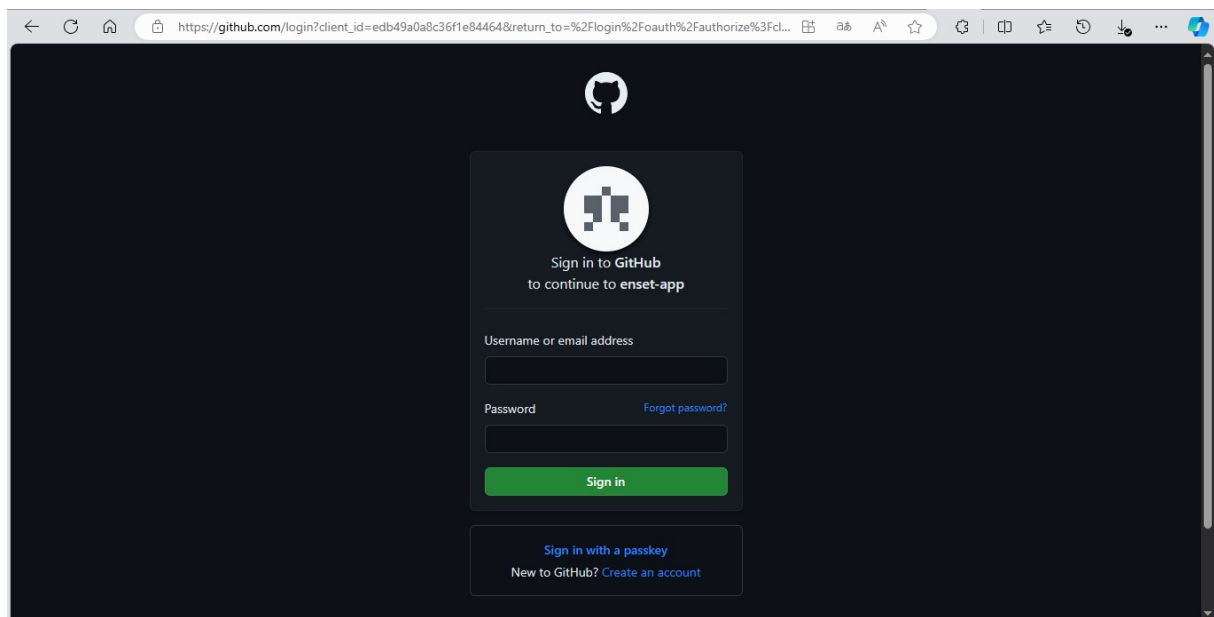


Google (USER)

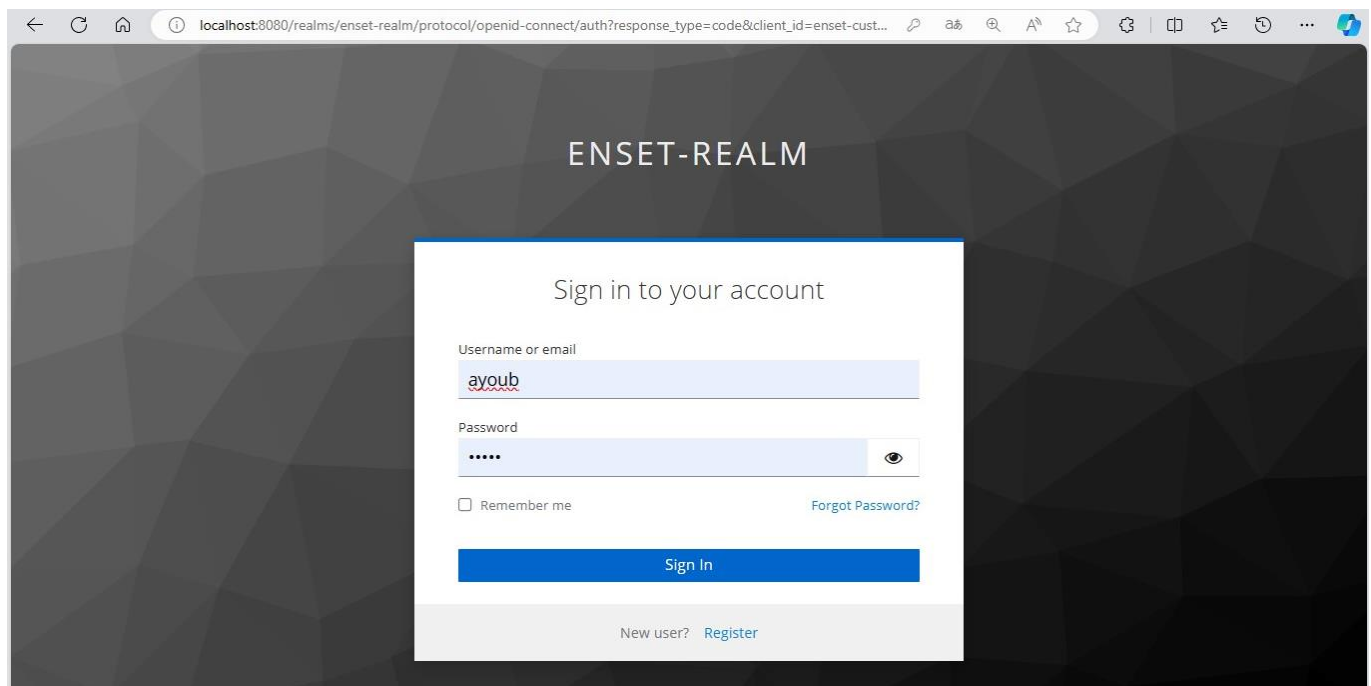




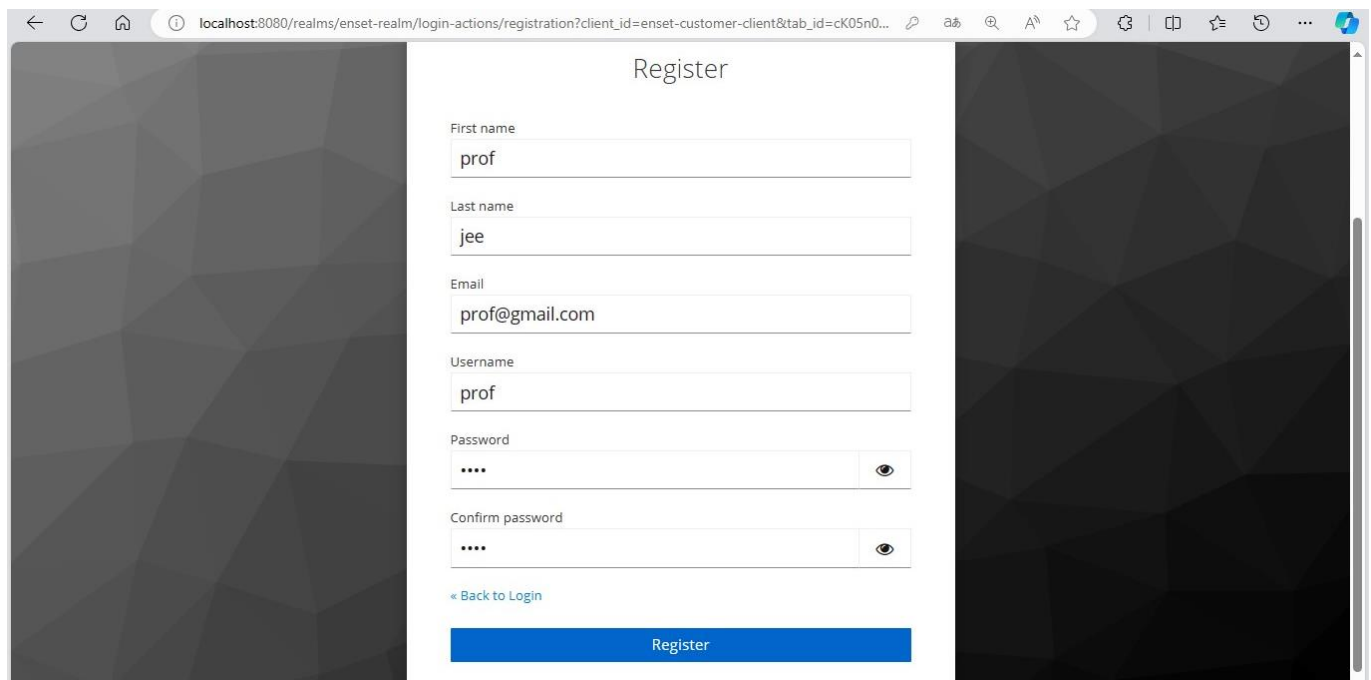
Github (USER)



keycloak



Ajouter user



Register

First name
prof

Last name
jee

Email
prof@gmail.com

Username
prof

Password
....

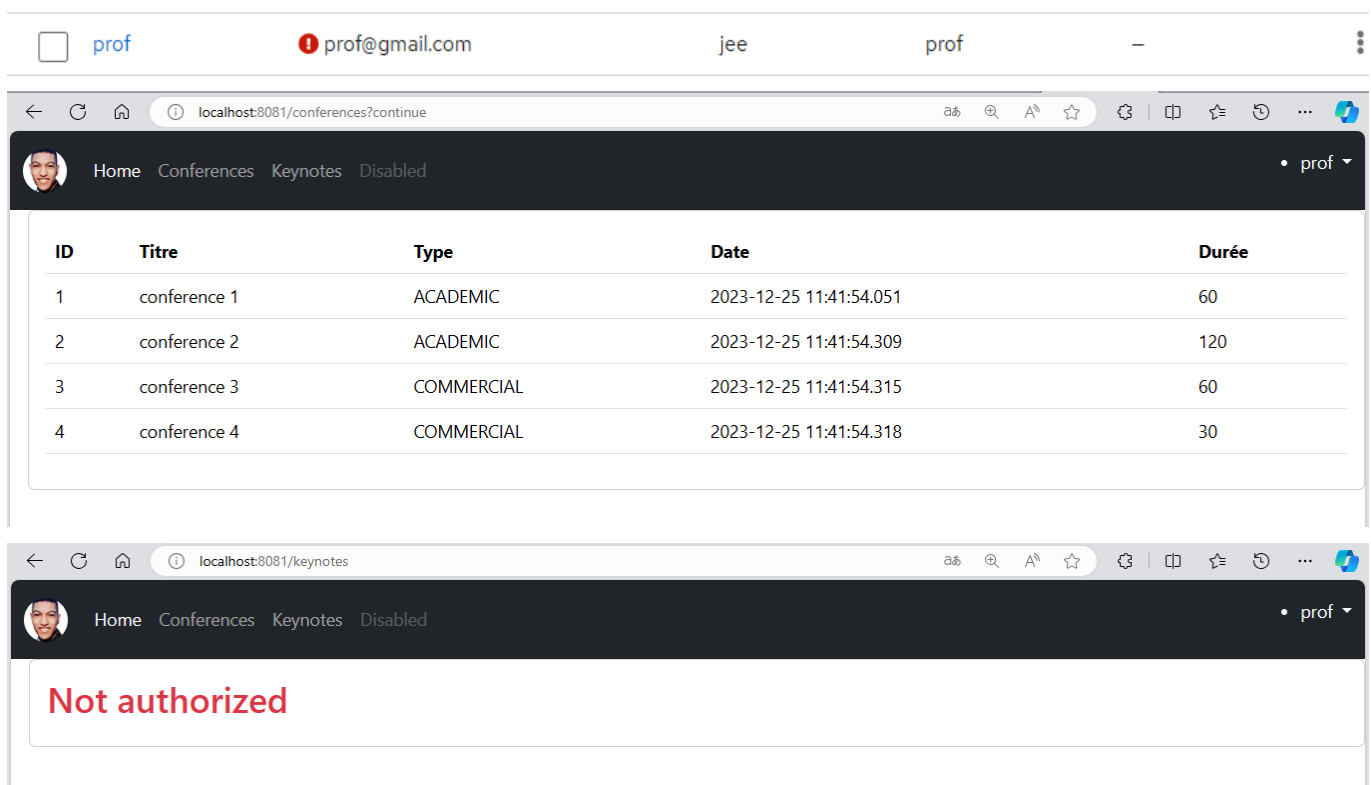
Confirm password
....

[« Back to Login](#)

Register

Keycloak

Prof (USER)



prof prof@gmail.com jee prof

localhost:8081/conferences?continue

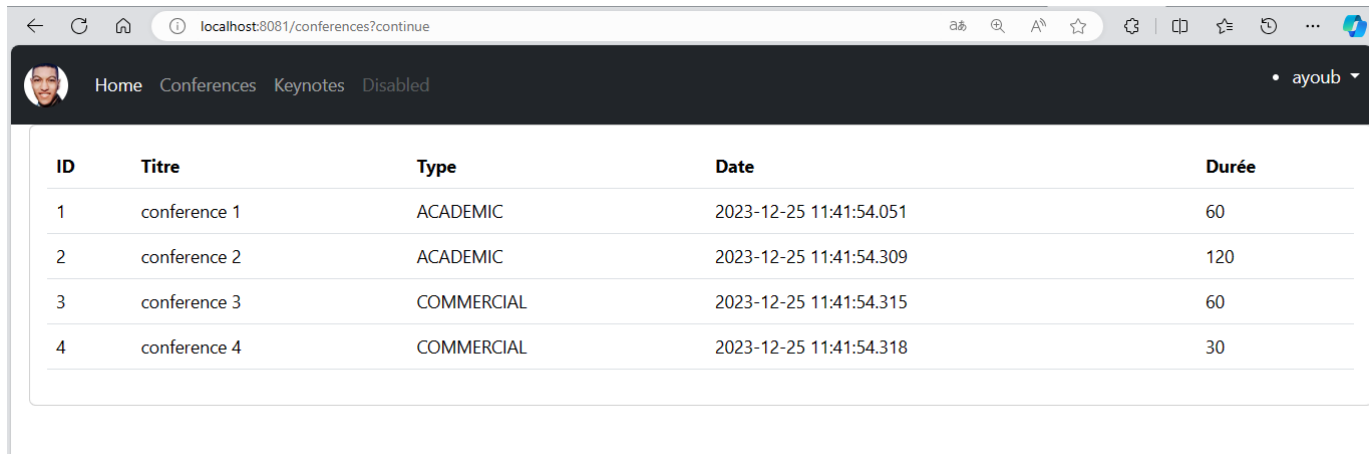
Home Conferences Keynotes Disabled • prof

ID	Titre	Type	Date	Durée
1	conference 1	ACADEMIC	2023-12-25 11:41:54.051	60
2	conference 2	ACADEMIC	2023-12-25 11:41:54.309	120
3	conference 3	COMMERCIAL	2023-12-25 11:41:54.315	60
4	conference 4	COMMERCIAL	2023-12-25 11:41:54.318	30

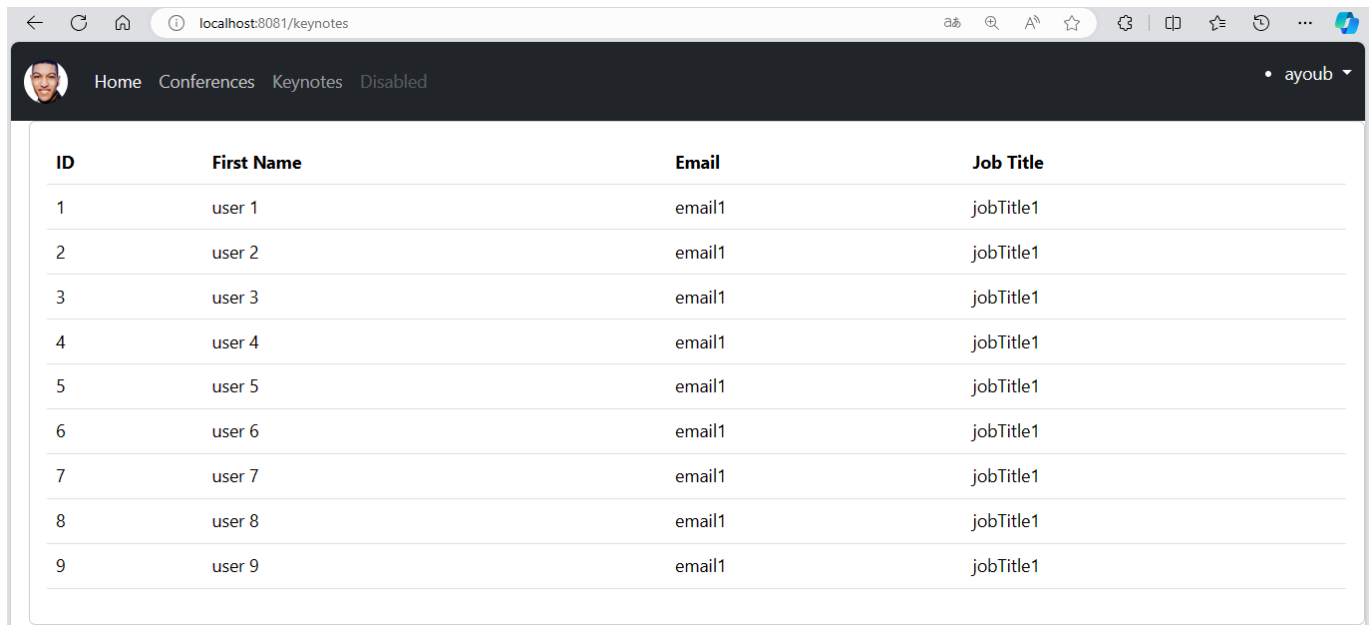
localhost:8081/keynotes

Home Conferences Keynotes Disabled • prof

Not authorized



ID	Titre	Type	Date	Durée
1	conference 1	ACADEMIC	2023-12-25 11:41:54.051	60
2	conference 2	ACADEMIC	2023-12-25 11:41:54.309	120
3	conference 3	COMMERCIAL	2023-12-25 11:41:54.315	60
4	conference 4	COMMERCIAL	2023-12-25 11:41:54.318	30



ID	First Name	Email	Job Title
1	user 1	email1	jobTitle1
2	user 2	email1	jobTitle1
3	user 3	email1	jobTitle1
4	user 4	email1	jobTitle1
5	user 5	email1	jobTitle1
6	user 6	email1	jobTitle1
7	user 7	email1	jobTitle1
8	user 8	email1	jobTitle1
9	user 9	email1	jobTitle1

H. Déployer l'application avec Docker et Docker compose

En cour

En conclusion, le traitement parallèle se dresse comme un pilier indispensable pour débloquer le potentiel du Big Data. Les défis posés par la gestion des données massives sont vastes, mais le traitement parallèle offre une réponse robuste, permettant de tirer parti de la puissance collective de multiples ressources. Des modèles de programmation tels que MapReduce ont ouvert la voie, tandis que des frameworks modernes tels que Spark et Flink ont apporté des améliorations significatives en termes de vitesse et de flexibilité. Les bases de données parallèles, les systèmes de fichiers distribués et l'utilisation judicieuse des GPU enrichissent encore davantage cette approche. À l'aube d'une ère où les données continuent de croître de manière exponentielle, le traitement parallèle en Big Data demeure une clé maîtresse, ouvrant la porte à une compréhension plus approfondie, à des insights plus rapides et à des découvertes qui façonnent l'avenir de l'informatique et de l'analyse des données.