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

Département Mathématique Informatique

Filière : Master-SDIA2

Modèle : Systèmes parallèle et Distribués

Rapport :

Examen Systèmes Distribués

Réalisé par :

- **Abdelkarim AGOUJIL**

Année Universitaire : 2022-2023

1. Introduction

On souhaite créer une application basée sur une architecture micro-service qui permet de gérer des réservations

concernant des ressources. Chaque réservation concerne une seule ressource. Une ressource est définie par son

id, son nom, son type (MATERIEL_INF0, MATERIEL_AUDIO_VUSUEL). Une réservation est définie par son id, son

nom, son contexte, , sa date, sa durée. Chaque réservation est effectuée par une personne. Une personne est

définie par son id, son nom, son email et sa fonction.

L'application doit permettre de gérer les ressources et les réservations. Pour faire plus simple, cette application

se composera de deux micro-services fonctionnels :

• Un Micro-service qui permet de gérer des « Resources-Service ».

• Un Micro-service qui permet de gérer les réservations effectuées par des personnes.

Les micro-services technique à mettre en place sont :

• Le service Gateway basé sur Spring cloud Gateway

• Le service Discovery base sur Eureka Server ou Consul Discovery (au choix)

• Le service de configuration basé sur Spring cloud config ou Consul Config (au choix)

Pour l'application, nous avons besoin de développer une frontend web, basé sur Angular Framework.

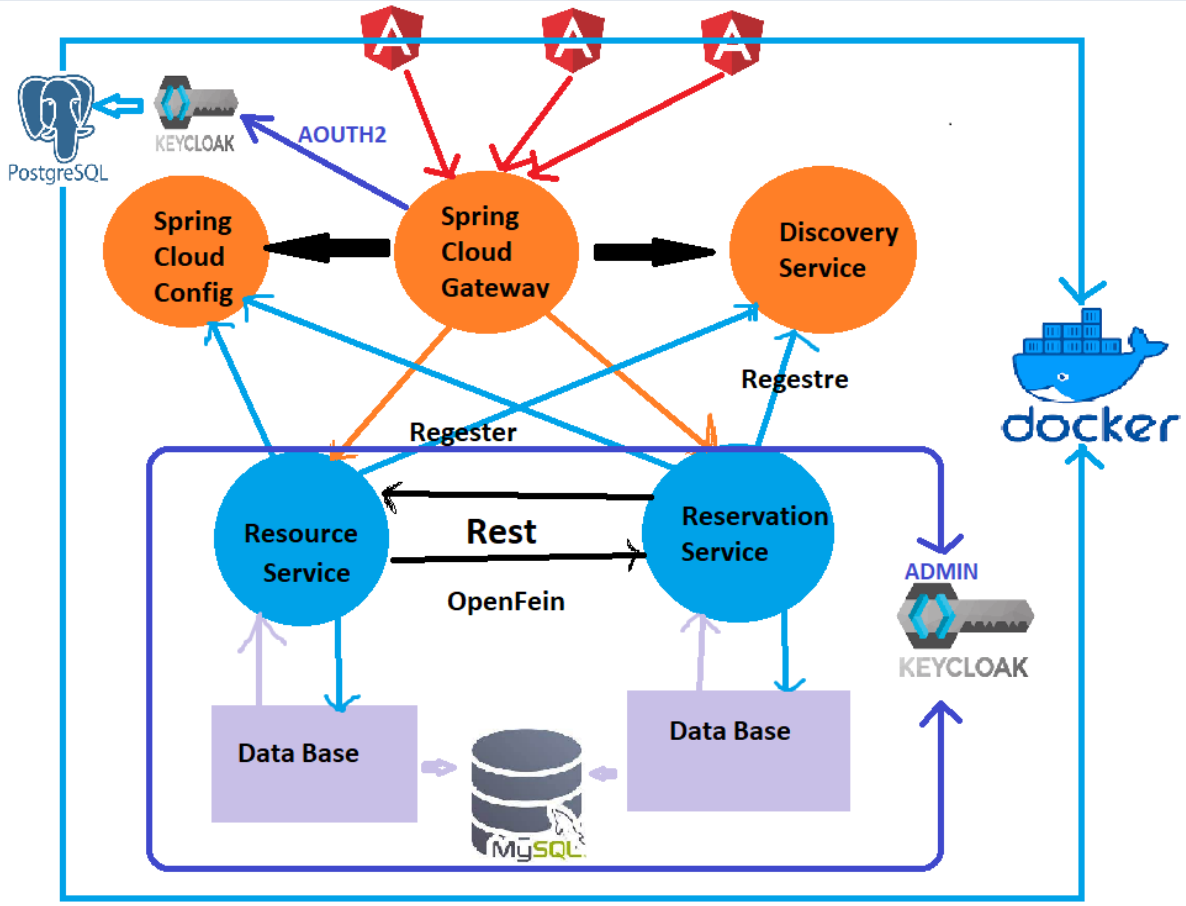
La sécurité de l'application est basée sur Oauth2 et OIDC avec Keycloak comme Provider

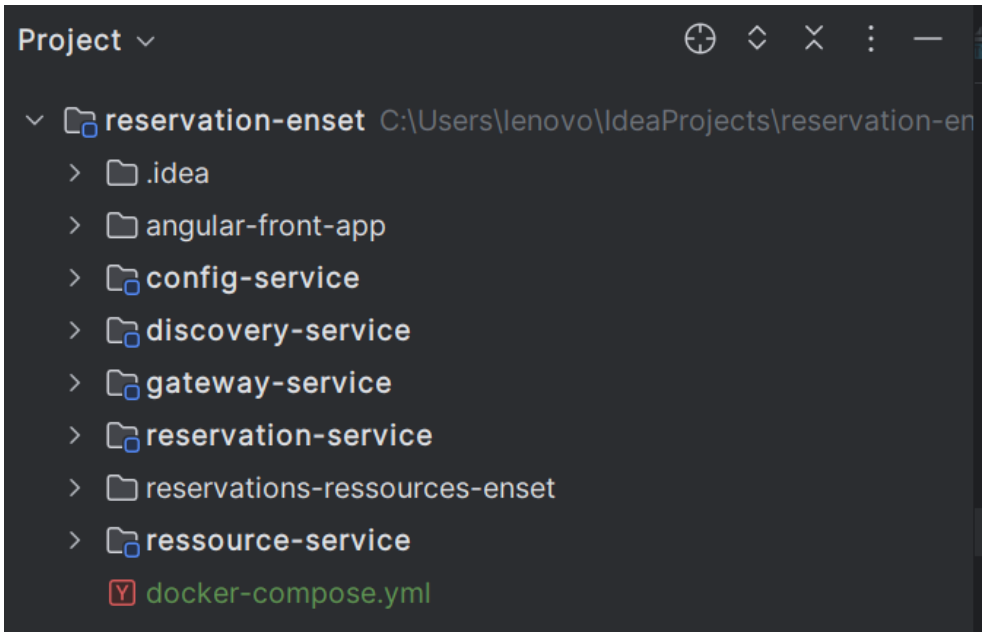
Pour les micro-services, il faut générer la documentation des web services Restfull en utilisant la spécification

OpenAPIDoc (Swagger). Prévoir aussi des circuit breakers basés sur Resilience4J comme solution de fault

tolerence

2. Architecture technique

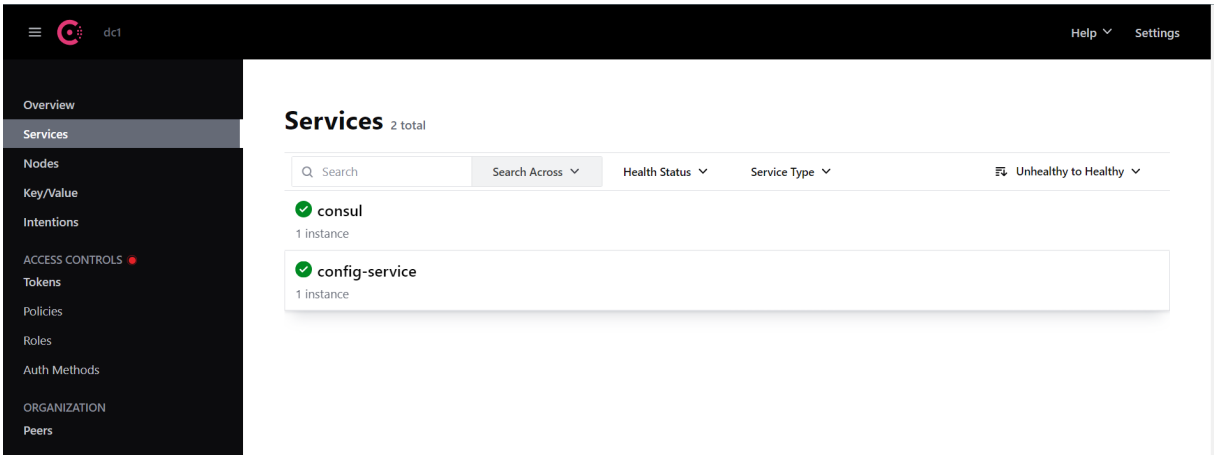




3. Développement du service config :

1) application.properties:

```
server.port=8888
spring.application.name=config-service
spring.cloud.config.server.git.uri=file:///C:/Users/lenovo/IdeaProjects/ms-enset-sdia/reservations-ressources-enset/config-repo
```



localhost:8888/ressource-service/default


```
{
  "name": "ressource-service",
  "profiles": [
    "default"
  ],
  "label": null,
  "version": "82d2bb3e7eaf1f7ad9edcf7906e65f71959adaec",
  "state": null,
  "propertySources": [
    {
      "name": "file:///C:/Users/lenovo/IdeaProjects/ms-enset-sdia/reservations-ressources-enset/config-repo/ressource-service.properties",
      "source": {
        "ressource.params.x": "345",
        "ressource.params.y": "367"
      }
    },
    {
      "name": "file:///C:/Users/lenovo/IdeaProjects/ms-enset-sdia/reservations-ressources-enset/config-repo/application.properties",
      "source": {
        "globale.params.p1": "123",
        "globale.params.p2": "12"
      }
    }
  ]
}
```


4. Développement du micro-service Ressource:

1. application.properties:

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

2. entités:
   A. Resource:

```
package ma.sdia.ressourceservice.entities;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.*;
import ma.sdia.ressourceservice.enums.ResourceType;
@Entity
@Getter @Setter @ToString @NoArgsConstructor @AllArgsConstructor @Builder
public class Ressource {
    @Id
```

```
    @GeneratedValue(strategy =
GenerationType.IDENTITY)
    private Long id;
    private String name;

    @Enumerated(EnumType.STRING)
    private ResourceType type;


}
```

2. enum:
   A. ResourceType:

```
package ma.sdia.ressourceservice.enums;


public enum ResourceType {
    MATERIEL_INFO, MATERIEL_AUDIO_VISUEL
}
```

3. repositories:
   B. ResourceType:

```
package ma.sdia.ressourceservice.repositories;


import ma.sdia.ressourceservice.entities.Ressource;
import
org.springframework.data.jpa.repository.JpaRepository
;
public interface ResourceRepository extends
JpaRepository<Ressource,Long> {
}
```

4. web:
   C. ResourceRestController:

```
package ma.sdia.ressourceservice.web;


import ma.sdia.ressourceservice.entities.Ressource;
import
ma.sdia.ressourceservice.repositories.ResourceReposit
```

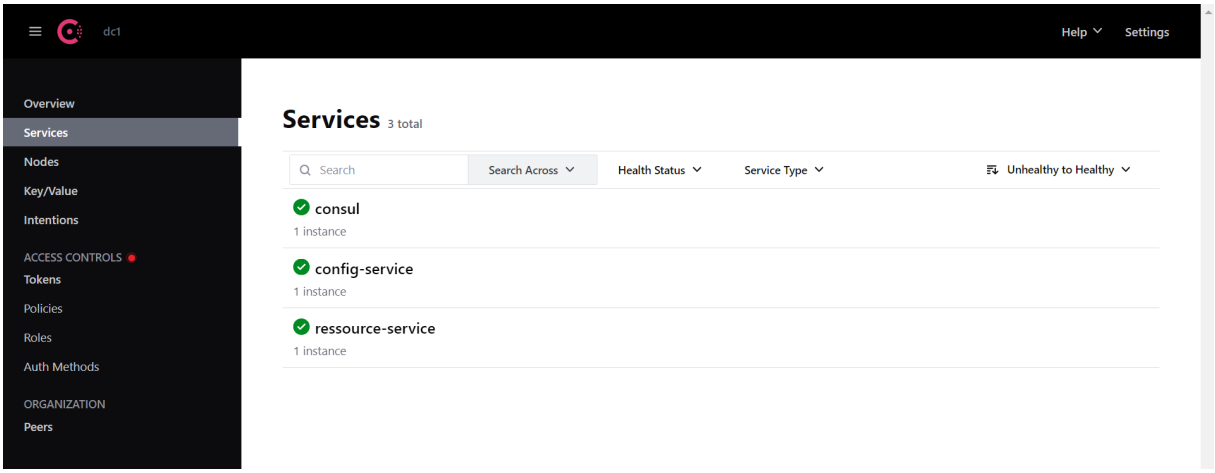
```
ory;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.PathVariable;
import
org.springframework.web.bind.annotation.RestControlle
r;


import java.util.List;


@RestController
public class RessourceRestController {
    ResourceRepository resourceRepository;
    public RessourceRestController(ResourceRepository
resourceRepository) {
        this.resourceRepository = resourceRepository;
    }
    @GetMapping("/ressources")
    public List<Ressource> ressourceList(){
        return resourceRepository.findAll();
    }
    @GetMapping("/ressources/{id}")
    public Ressource resourceById(@PathVariable Long
id){
        return resourceRepository.findById(id).get();
    }


}
```

5. application:
   D. RessourceServiceApplication:

```
package ma.sdia.ressourceservice;


import ma.sdia.ressourceservice.entities.Ressource;
```

```
import ma.sdia.ressourceservice.enums.ResourceType;
import
ma.sdia.ressourceservice.repositories.ResourceReposit
ory;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootAppl
ication;
import org.springframework.context.annotation.Bean;

import java.util.List;

@SpringBootApplication
public class RessourceServiceApplication {

  public static void main(String[] args) {

SpringApplication.run(RessourceServiceApplication.cla
ss, args);
  }



  @Bean
  CommandLineRunner
commandLineRunner(ResourceRepository
resourceRepository){
     return args -> {

        List<Ressource> customerList=List.of(
              Ressource.builder()
                    .name("Resource1")
                    .type(ResourceType.MATERIEL_INFO)
                    .build(),
```

```
                Ressource.builder()
                        .name("Resource2")

.type(ResourceType.MATERIEL_AUDIO_VISUEL)
                        .build()
            );
            resourceRepository.saveAll(customerList);
        };
    }

}
```



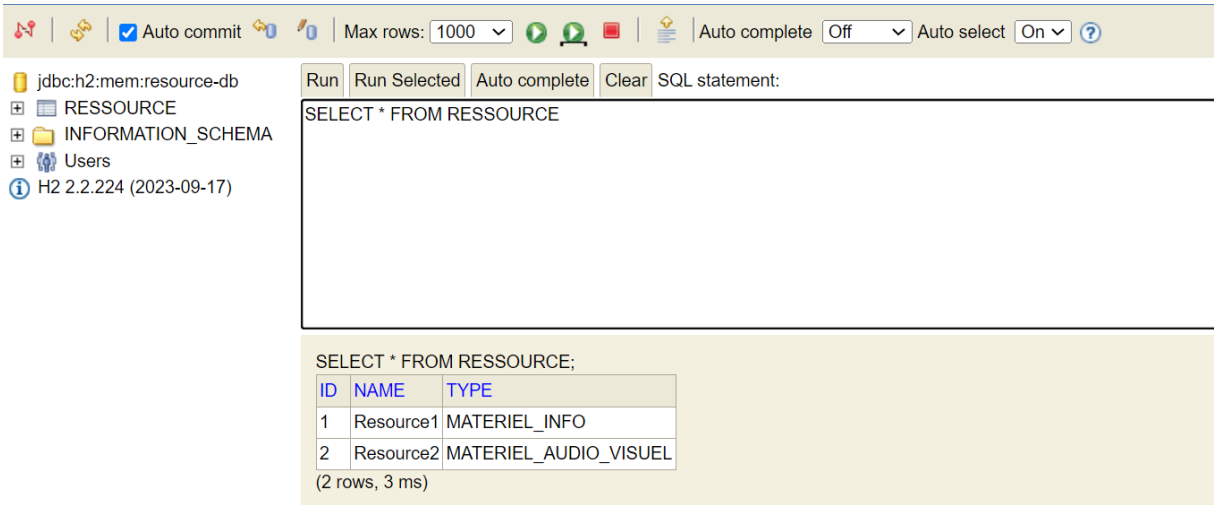
localhost:8081/ressources


```
[
  {
    "id": 1,
    "name": "Resource1",
    "type": "MATERIEL_INFO"
  },
  {
    "id": 2,
    "name": "Resource2",
    "type": "MATERIEL_AUDIO_VISUEL"
  }
]
```


localhost:8081/ressources/1


```
{
  "id": 1,
  "name": "Resource1",
  "type": "MATERIEL_INFO"
}
```


```
management.endPoints.web.exposure.include=*
spring.datasource.url=jdbc:h2:mem:resource-db
spring.h2.console.enabled=true
```



6. Développement du service gateway :

3. Application:

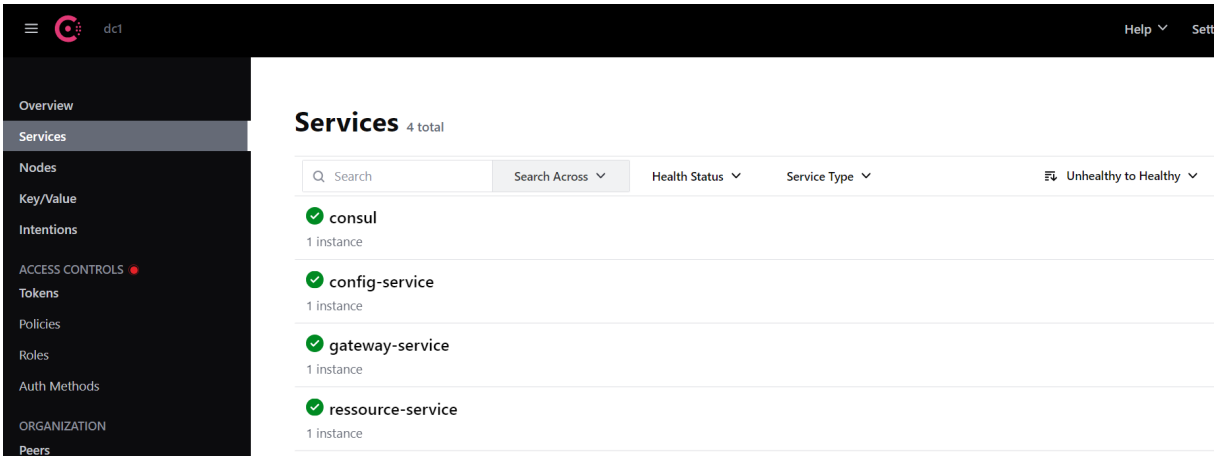
```
package ma.sdia.gatewayservice;


import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootAppl
ication;
import
org.springframework.cloud.client.discovery.ReactiveDi
scoveryClient;
```

```
import
org.springframework.cloud.gateway.discovery.Discovery
ClientRouteDefinitionLocator;
import
org.springframework.cloud.gateway.discovery.Discovery
LocatorProperties;
import org.springframework.context.annotation.Bean;


@SpringBootApplication
public class GatewayServiceApplication {


  public static void main(String[] args) {

SpringApplication.run(GatewayServiceApplication.class
, args);
  }


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





}
```

2) application.properties:

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

4)Test:



7. Développement du micro-service Reservation :

4. Entités JPA et Interface JpaRepository basées sur Spring data

a) Entité Reservation

```
package ma.sdia.reservationservice.entities;

import jakarta.persistence.*;
import lombok.*;
import ma.sdia.reservationservice.model.Ressource;

import java.util.Date;

@Entity
@Getter @Setter @AllArgsConstructor @NoArgsConstructor
@Builder
public class Reservation {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;

    private String context;

    private Date date;
```

```
    private int duration;
    @Transient
    private Ressource ressource;
    private Long ressourceId;
    @Transient
    private Personne personne;
    private String personneId;



}
```

b) **Entité Personne**

```
package ma.sdia.reservationservice.entities;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.*;

@Entity
@Getter @Setter @AllArgsConstructor @NoArgsConstructor
@Builder
public class Personne {
    @Id
    private String id;
    private String name;
    private String email;
    private String fonction;
}
```

c) **model:**

```
package ma.sdia.reservationservice.model;

public enum ResourceType {
    MATERIEL_INFO, MATERIEL_AUDIO_VISUEL
}
```

```
package ma.sdia.reservationservice.model;

import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.*;

@Getter @Setter @ToString @NoArgsConstructor
@AllArgsConstructor @Builder
public class Ressource {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    private String type;

}
```

d) **PersonneRepository:**

```
package ma.sdia.reservationservice.repositories;

import ma.sdia.reservationservice.entities.Personne;
import
org.springframework.data.jpa.repository.JpaRepository;
```

```
public interface PersonneRepository extends
JpaRepository<Personne,String> {

}
```

e) **ReservationRepositories:**

```
package ma.sdia.reservationservice.repositories;


import
ma.sdia.reservationservice.entities.Reservation;
import
org.springframework.data.jpa.repository.JpaRepository;


public interface ReservationRepository extends
JpaRepository<Reservation,Long> {

}
```

f) **RessourceRestClient**

```
package ma.sdia.reservationservice.web;


import ma.sdia.reservationservice.model.Ressource;
import
org.springframework.cloud.openfeign.FeignClient;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.PathVariable;


import java.util.List;


@FeignClient("ressource-service")
public interface ResourceRestClient {
    @GetMapping("/ressources")
    public List<Ressource> allRessource();
    @GetMapping("/ressources/{id}")
```

```
    public Ressource findRessourceById(@PathVariable
Long id);


}
```

g) ReservatinRestController

```
package ma.sdia.reservationservice.web;


import ma.sdia.reservationservice.entities.Personne;
import
ma.sdia.reservationservice.entities.Reservation;
import ma.sdia.reservationservice.model.Ressource;
import
ma.sdia.reservationservice.repositories.PersonneReposi
tory;
import
ma.sdia.reservationservice.repositories.ReservationRep
ository;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.Date;
import java.util.List;
import java.util.Random;

@RestController
public class ReservationRestController {
    private ReservationRepository
reservationRepository;
    private ResourceRestClient resourceRestClient;
    private PersonneRepository personneRepository;
    public
ReservationRestController(ReservationRepository
```

```
reservationRepository, ResourceRestClient
resourceRestClient, PersonneRepository
personneRepository) {
        this.reservationRepository =
reservationRepository;
        this.resourceRestClient = resourceRestClient;
        this.personneRepository = personneRepository;
    }

    @GetMapping("/reservations")
    public List<Reservation> reservationList(){
        List<Reservation> reservationList =
reservationRepository.findAll();
        reservationList.forEach(rserv->{
            rserv.setPersonne(
personneRepository.findById(rserv.getPersonneId()).get
());

rserv.setRessource(resourceRestClient.findRessourceByI
d(rserv.getRessourceId()));
        });
        return reservationList;
    }
    @GetMapping("/reservation/{id}")
    public Reservation reservationById(@PathVariable
Long id){
        Reservation reservation=
reservationRepository.findById(id).get();
        Ressource
ressource=resourceRestClient.findRessourceById(reserva
tion.getRessourceId());
        reservation.setRessource(ressource);
        return reservation;
    }
    @GetMapping("/users")
```

```
    public List<Personne> personneList(){
        List<Personne>
personneList=personneRepository.findAll();
        return personneList;
    }
    @GetMapping("/users/{id}")
    public Personne personneById(@PathVariable String
id) {
        Personne p =
personneRepository.findById(id).get();
        return p;
    }

    @PostMapping("/reserve")
    public ResponseEntity<Reservation>
createReservation(@RequestBody Reservation
reservation) {

System.out.println("--------------------------------")
;

System.out.println(reservation.getPersonneId());

System.out.println("--------------------------------")
;
        Personne personne =
personneRepository.findById(reservation.getPersonneId(
)).orElse(null);
        Ressource ressource =
resourceRestClient.findRessourceById(reservation.getRe
ssourceId());

        if (personne != null && ressource != null) {
            reservation.setPersonne(personne);
            reservation.setRessource(ressource);
            reservation.setName("reservation"+new
```

```
Random().nextInt(100));
            reservation.setDate(new Date());
            reservation.setContext("Informatique");
            reservation.setDuration(new
Random().nextInt(2, 30));
            Reservation createdReservation =
reservationRepository.save(reservation);


            return new
ResponseEntity<>(createdReservation,
HttpStatus.CREATED);
        } else {
            return new
ResponseEntity<>(HttpStatus.BAD_REQUEST);
        }
    }
    @PostMapping("/addUser")
    public ResponseEntity<Personne>
createPersonne(@RequestBody Personne p) {
        Personne personne =
personneRepository.findById(p.getId()).orElse(null);
        if (personne == null ) {
            Personne personne1= Personne.builder()
                    .id(p.getId())
                    .name(p.getName())
                    .email(p.getEmail())
                    .fonction("Unknow")
                    .build();
            Personne createdPersonne =
personneRepository.save(personne1);


            return new
ResponseEntity<>(createdPersonne, HttpStatus.CREATED);
        } else {
```

```
            return new
ResponseEntity<>(HttpStatus.BAD_REQUEST);
        }
    }
}
```

h) ReservationServiceApplication

```
package ma.sdia.reservationservice;


import ma.sdia.reservationservice.entities.Personne;
import
ma.sdia.reservationservice.entities.Reservation;
import ma.sdia.reservationservice.model.Ressource;
import
ma.sdia.reservationservice.repositories.PersonneReposi
tory;
import
ma.sdia.reservationservice.repositories.ReservationRep
ository;
import
ma.sdia.reservationservice.web.ResourceRestClient;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootAppli
cation;
import
org.springframework.cloud.openfeign.EnableFeignClients
;
import org.springframework.context.annotation.Bean;
import java.util.Date;
import java.util.List;
import java.util.Random;
import java.util.UUID;
```

```
@SpringBootApplication
@EnableFeignClients
public class ReservationServiceApplication {

  public static void main(String[] args) {

SpringApplication.run(ReservationServiceApplication.class, args);
  }

  @Bean
  CommandLineRunner commandLineRunner(ReservationRepository reservationRepository, ResourceRestClient resourceRestClient, PersonneRepository personneRepository) {
     return args -> {
        Personne personne1=new Personne(UUID.randomUUID().toString(),"Abdelkarim","abdlkrim@gmail.com","Etudiant");
        Personne personne2=new Personne(UUID.randomUUID().toString(),"Agoujil","agoujil@gmail.com","Etudiant");
        personneRepository.save(personne1);
        personneRepository.save(personne2);


List<Ressource>ressourceList=resourceRestClient.allRessource();


        personneRepository.findAll().forEach(p -> {
           Reservation reservation1 = Reservation.builder()
                  .name("reservation1")
                  .context("context1")
                  .date(new Date())
```

```
                    .duration(new Random().nextInt(2,
30))

.ressourceId(ressourceList.get(0).getId())
                    .personne(p)
                    .personneId(p.getId())
                    .build();
            Reservation reservation2 =
Reservation.builder()
                    .name("reservation2")
                    .context("context2")
                    .date(new Date())
                    .duration(new Random().nextInt(2,
30))

.ressourceId(ressourceList.get(1).getId())
                    .personne(p)
                    .personneId(p.getId())
                    .build();
            reservationRepository.save(reservation1);
            reservationRepository.save(reservation2);
        });


    };


  }
}
```

i) Teste:

localhost:9999/RESERVATION-SERVICE/reservations

Gmail YouTube Maps


```
    "id": 1,
    "name": "reservation1",
    "context": "context1",
    "date": "2024-01-01T16:55:39.017+00:00",
    "duration": 12,
    "ressource": {
        "id": 1,
        "name": "Resource1",
        "type": "MATERIEL_INFO"
    },
    "ressourceId": 1,
    "personne": {
        "id": "30feefa6-86f0-40d1-9deb-7851209f8161",
        "name": "Abdelkarim",
        "email": "abdlkrim@gmail.com",
        "fonction": "Etudiant"
    },
    "personneId": "30feefa6-86f0-40d1-9deb-7851209f8161"
},
{
    "id": 2,
    "name": "reservation2",
    "context": "context2",
    "date": "2024-01-01T16:55:39.017+00:00",
    "duration": 10,
```


8. Développer un simple frontend web pour l'application:

Navbar Home Resources • khalid

Welcom to Agoujil web site

Navbar Home Resources Reservations Users • agoujil

ID	Name	Type	
1	Resource1	MATERIEL_INFO	Reserve
2	Resource2	MATERIEL_AUDIO_VISUEL	Reserve

Navbar Home Resources Reservations Users • agoujil

ID	Name	Context	Date	Duration	Ressource Id	Personne Id	
1	reservation1	context1	2024-01-01T14:40:24.126+00:00	16	1	0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0	Delete
2	reservation2	context2	2024-01-01T14:40:24.126+00:00	26	2	0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0	Delete
3	reservation1	context1	2024-01-01T14:40:24.153+00:00	24	1	43fb0c9e-290e-4c83-b320-0ae1367fd4a7	Delete
4	reservation2	context2	2024-01-01T14:40:24.153+00:00	24	2	43fb0c9e-290e-4c83-b320-0ae1367fd4a7	Delete
5	reservation70	Informatique	2024-01-01T14:41:40.463+00:00	5	1	6d6a390a-c02b-4c73-b819-17b062851f8f	Delete

Navbar Home Resources Reservations Users • agoujil

ID	Name	Email	Fonction		
0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0	Abdelkarim	abdlkrim@gmail.com	Etudiant	Delete	Update
43fb0c9e-290e-4c83-b320-0ae1367fd4a7	Agoujil	agoujil@gmail.com	Etudiant	Delete	Update
6d6a390a-c02b-4c73-b819-17b062851f8f	khalid	khalid@gmail.com	Unknow	Delete	Update
e8785a80-f89a-478a-b4bc-f0e638c70d5c	agoujil	agoujil@gmail.com	Unknow	Delete	Update

9. Sécuriser l'application avec une authentification Keycloak:

KEYCLOAK agoujil

Create realm

A realm manages a set of users, credentials, roles, and groups. A user belongs to and logs into a realm. Realms are isolated from one another and can only manage and authenticate the users that they control.

Resource file: Drag a file here or browse to upload | Browse... | Clear

1

Upload a JSON file

Realm name * reservation-realm

KEYCLOAK agoujil

Clients > Create client

Create client

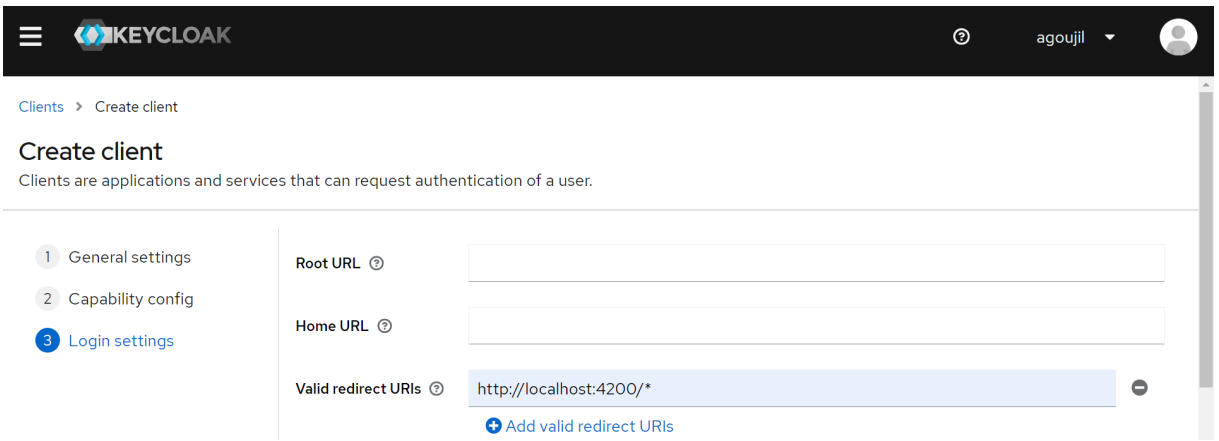
Clients are applications and services that can request authentication of a user.

1 General settings
2 Capability config
3 Login settings

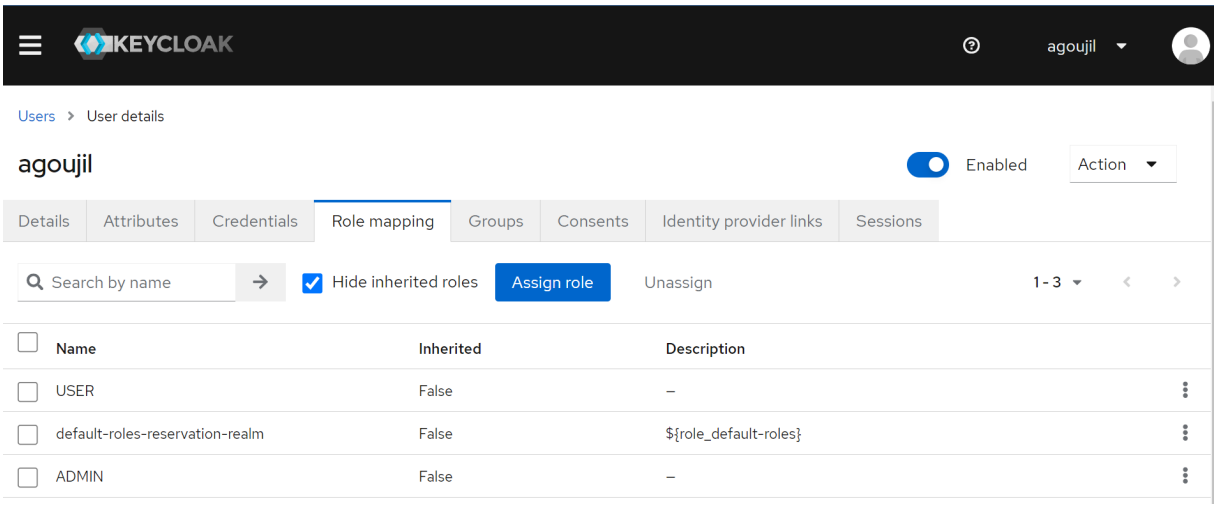
Client type: OpenID Connect

Client ID * sdia-reservation-client

Name


KEYCLOAK
agoujil
Clients > Create client
Create client
Clients are applications and services that can request authentication of a user.
1 General settings
2 Capability config
3 Login settings
Root URL
Home URL
Valid redirect URIs
http://localhost:4200/*
Add valid redirect URIs


KEYCLOAK
agoujil
Create user
Required user actions
Select action
Username *
agoujil
Email
agoujil@gmail.com
Email verified
No
First name
ABDELKARIM
Last name
AGOUJIL


KEYCLOAK
agoujil
Users > User details
agoujil
Enabled
Action
Details
Attributes
Credentials
Role mapping
Groups
Consents
Identity provider links
Sessions
Search by name
Hide inherited roles
Assign role
Unassign
1 - 3
Name
Inherited
Description
USER
False
–
default-roles-reservation-realm
False
${role_default-roles}
ADMIN
False
–

GET Untitled Request | POST http://localhost:8080/re | No Environment

http://localhost:8080/realms/sdia-realm/protocol/openid-connect/token

Save

POST http://localhost:8080/realms/sdia-realm/protocol/openid-connect/token Send

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

☑	username	agoujil
☑	password	1234

Body Cookies Headers (12) Test Results 200 OK 396 ms 2.99 KB Save as example

Pretty Raw Preview Visualize JSON


```
1 {"access_token":"eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJxN2RtWXR4U0JwcC12VGZCUGJXb2QtU1pjS09NW
mZCRDZHSVhoZEstZnhFIn0.
eyJleHAiOjE3MDQwNTY4OTUsImlhdCI6MTcwNDA1NjU5NSwianRpIjoiMjQ2ZTE4ZDQtODc1OC00M2IyLWIxMmEtNDI4ZmY4MzkxNT
ZhIiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4MDgwL3JlYWxtcy9zZGlhLXJlYWxtIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6ImU4
Nzg1YTgwLWY4OWEtNDc4YS1iNGJjLWYwZTYzOGM3MGQ1YyIsInR5cCI6IkJlYXJlciIsImF6cCI6InNkaWEtcmVzZXJ2YXRpb24tY2
xpZW50Iiwic2Vzc2lvbl9zdGF0ZSI6IjAxNzFmM2YyLTA2NDgtNDgwNC1iOWEzLTQ4YzdiYjQzNGMyYiIsImFjciI6IjEiLCJhbGxv
d2VkLW9yaWdpbnMiOlsiaHR0cDovL2xvY2FsaG9zdDo0MjAwIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJkZWZhdWx0LXJvbG
VzLXNkaWEtcmVhbG0iLCJvZmZsaW5lX2FjY2VzcyIsInVtYV9hdXRob3JpemF0aW9uIiwiQURNSU4iLCJVU0VSI119LCJyZXNvdXJj
ZV9hY2Nlc3MiOnsiYWNjb3VudCI6eyJyb2xlcyI6WyJtYW5hZ2UtYWNjb3VudCIsIm1hbmFnZS1hY2NvdW50LWxpbmtzIiwidmlldy
```

JWT

Debugger Libraries Introduction Ask

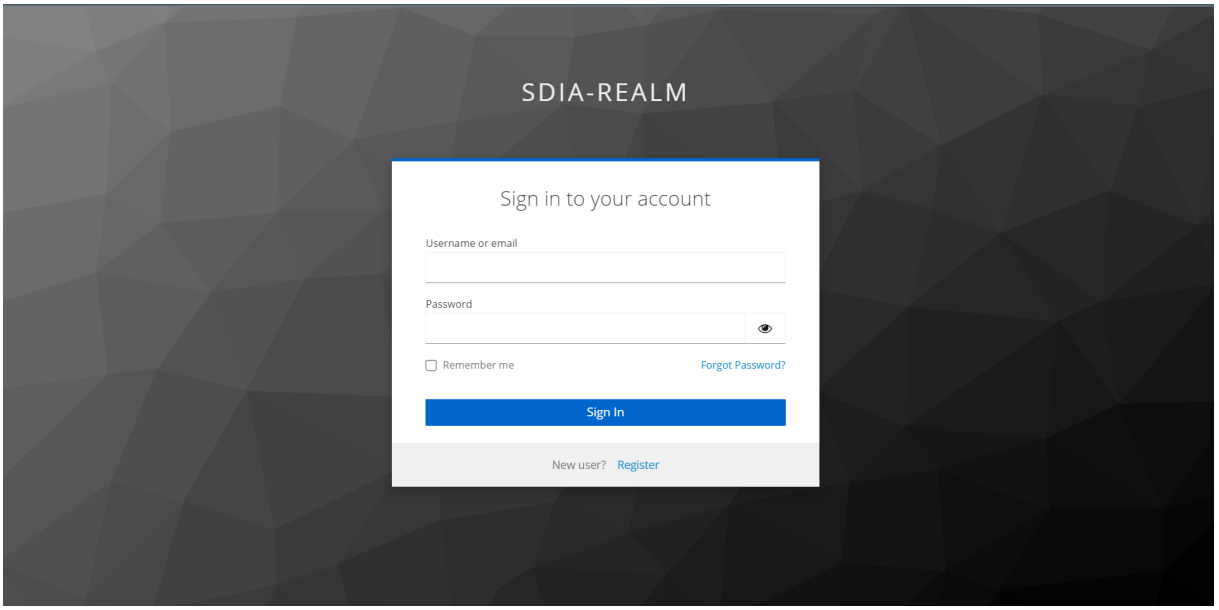
Crafted by Auth0 by Okta

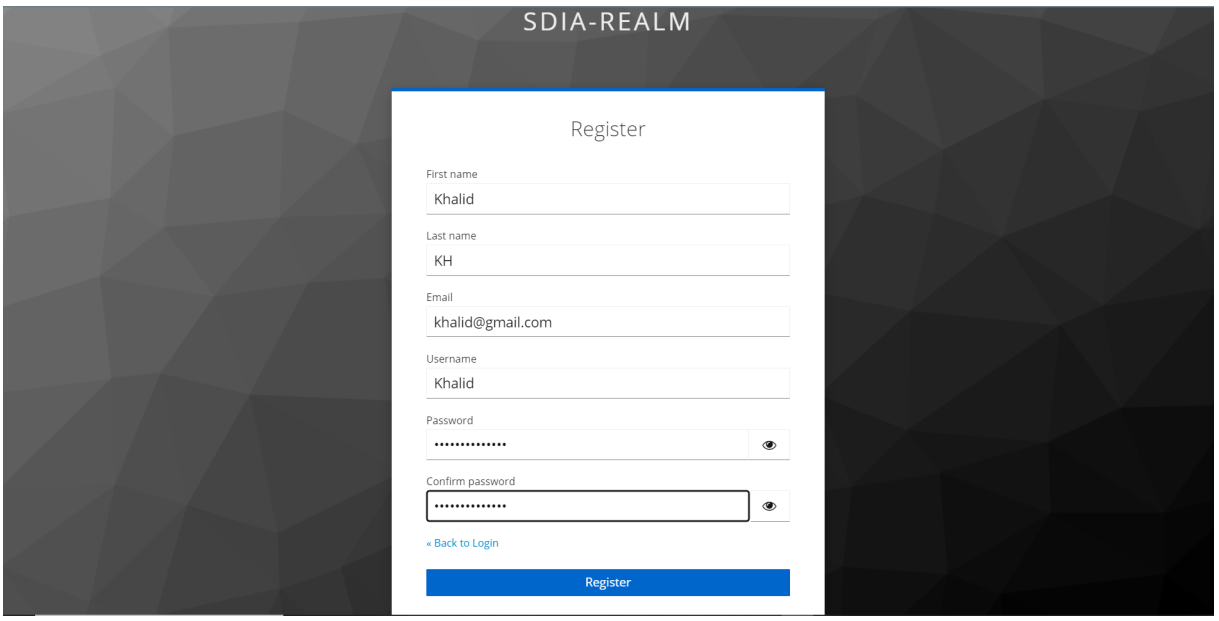
eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJxN2RtWXR4U0JwcC12VGZCUGJXb2QtU1pjS09NWmZCRDZHSVhoZEstZnhFIn0.eyJleHAiOjE3MDQwNTY4OTUsImlhdCI6MTcwNDA1NjU5NSwianRpIjoiMjQ2ZTE4ZDQtODc1OC00M2IyLWIxMmEtNDI4ZmY4MzkxNTZhIiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4MDgwL3JlYWxtcy9zZGlhLXJlYWxtIiwiYXVkIjoiYWNjb3VudCIsInN1YiI6ImU4Nzg1YTgwLWY4OWEtNDc4YS1iNGJjLWYwZTYzOGM3MGQ1YyIsInR5cCI6IkJlYXJlciIsImF6cCI6InNkaWEtcmVzZXJ2YXRpb24tY2xpZW50Iiwic2Vzc2lvbl9zdGF0ZSI6IjAxNzFmM2YyLTA2NDgtNDgwNC1iOWEzLTQ4YzdiYjQzNGMyYiIsImFjciI6IjEiLCJhbGxvd2VkLW9yaWdpbnMiOlsiaHR0cDovL2xvY2FsaG9zdDo0MjAwIl0sInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJkZWZhdWx0LXJvbGVzLXNkaWEtcmVhbG0iLCJvZmZsaW5lX2FjY2VzcyIsInVtYV9hdXRob3JpemF0aW9uIiwiQURNSU4iLCJVU0VSIl19LCJyZXNvdXJjZV9hY2Nlc3MiOnsiYWNjb3VudCI6eyJyb2xlcyI6WyJtYW5hZ2UtYWNjb3VudCIsIm1hbmFnZS1hY2NvdW50LWxpbmtzIiwidmlldy1wcm9maWxlIl19fSwic2NvcGUiOiJlbWFpbCBwcm9maWxlIiwic2lkIjoiMDE3MWYzZjItMDY0OC00ODA0LWI5YTMtNDhjN2JiNDM0YzJiIiwiZW1haWxfdmVyaWZpZWQiOmZhbHNlLCJuYW1lIjoiQWJkZWxrYXJpbSBBZ291amlsIiwicHJlZmVycmVkX3VzZXJuYW1lIjoiYWdvdWppbCIsImdpdmVuX25hbWUiOiJBYmRlbGthcmltIiwiZmFtaWx5X25hbWUiOiJBZ291amlsIiwiZW1haWwiOiJhZ291amlsQGdtYWlsLmNvbSJ9.pTMF2fycoup5YN1g7MVzp3BnS0E8EfRE3Xdcdb8X2Z6dhkVfN2I-NlRzqtPoYIphzLW2rlQF56nf5tvzyBS1tOvgD67y5kA_-N9hAjdBYKUhmuuIp92s_-ofkqOdUOTg9VAVwETI36axNZ2xCXs1HoZ5ZsFcvJPsDQY4BnAc1FLZ1H9iR2g-_5gcHYZ7d5X2kq6puwTE_yYAAs4GpxM7LUgW31y


```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "q7dmYtxSBpp-vTfBPbWod-SZcKOMZfBD6GIXhdK-fxE"
}
```

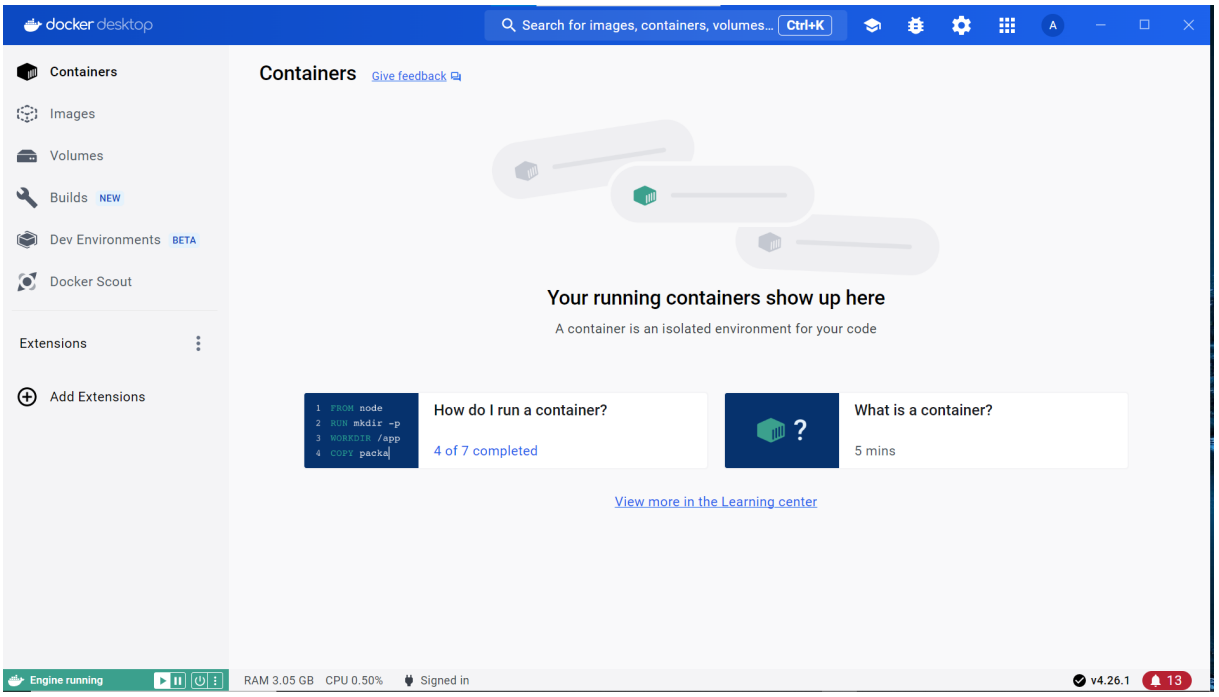
PAYLOAD: DATA

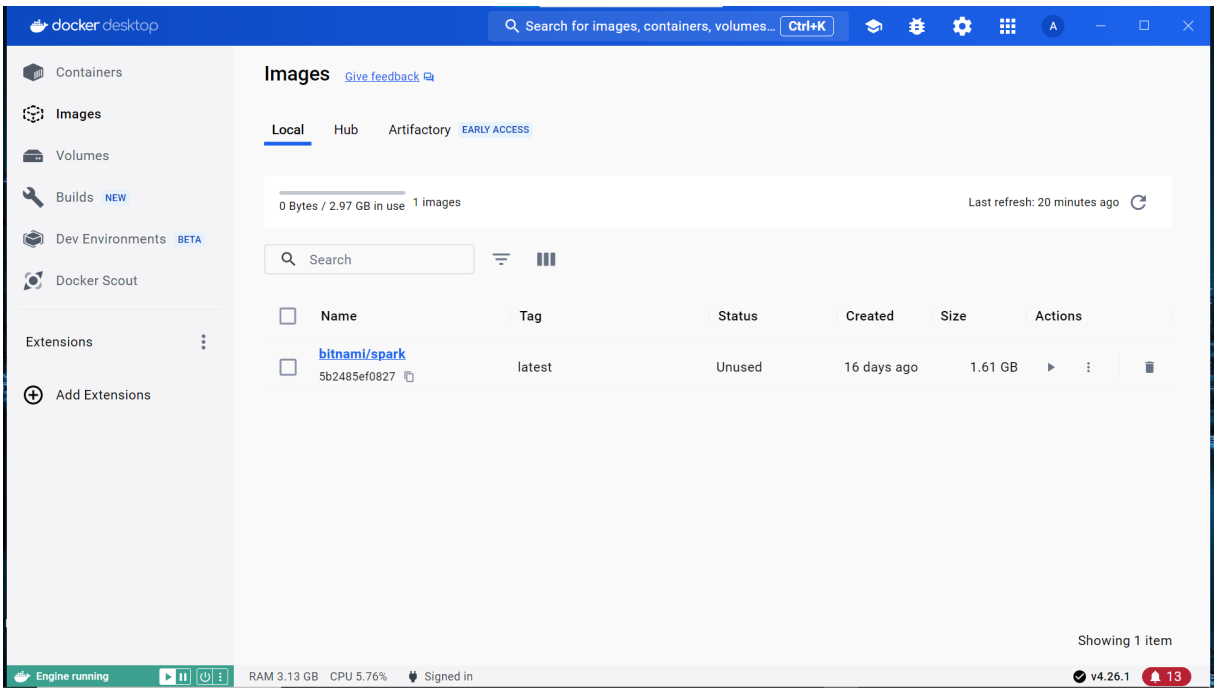
```
{
  "exp": 1704056895,
  "iat": 1704056595,
  "jti": "246e18d4-8758-43b2-b12a-428ff839156a",
  "iss": "http://localhost:8080/realms/sdia-realm",
  "aud": "account",
  "sub": "e8785a80-f89a-478a-b4bc-f0e638c70d5c",
  "typ": "Bearer",
  "azp": "sdia-reservation-client",
  "session_state": "0171f3f2-0648-4804-b9a3-
48c7bb434c2b",
  "acr": "1",
  "allowed-origins": [
    "http://localhost:4200"
  ],
  "realm_access": {
    "roles": [
      "default-roles-sdia-realm",
      "offline_access",
      "uma_authorization",
      "ADMIN",
      "USER"
    ]
  },
  "resource_access": {
    "account": {
      "roles": [
        "manage-account",
        "manage-account-links",
        "view-profile"
      ]
    }
  },
  "scope": "email profile",
  "sid": "0171f3f2-0648-4804-b9a3-48c7bb434c2b",
  "email_verified": false,
  "name": "Abdelkarim Agoujil",
  "preferred_username": "agoujil",
  "given_name": "Abdelkarim",
  "family_name": "Agoujil",
  "email": "agoujil@gmail.com"
}
```



10. Déployer l'application avec Docker et Docker compose :





docker-compose.yml:

```yaml
services:
 discovery-service:
   build: ./discovery-service
   container_name: discovery-service
   restart: always
   ports:
     - '8761:8761'
   expose:
     - '8761'

 config-service:
   build: ./config-service
   container_name: config-service
   restart: always
   ports:
     - '8888:8888'
```

```
    expose:
      - '8888'
    environment:
      - DISCOVERY_SERVICE_URL=http://discovery-service:8761/eureka
    depends_on:
      - discovery-service

  gateway-service:
    build: ./gateway-service
    container_name: gateway-service
    restart: always
    ports:
      - '9999:9999'
    expose:
      - '9999'
    environment:
      - DISCOVERY_SERVICE_URL=http://discovery-service:8761/eureka
      - CONFIG_SERVER=http://config-service:9999
    depends_on:
      - config-service

  postgres-keycloak-db:
    image: postgres
```

```
    container_name: postgres-keycloak-db
    volumes:
      -
postgres_keycloak_data_ex:/var/lib/postgr
esql/data
    environment:
      POSTGRES_DB: keycloak_db
      POSTGRES_USER: admin
      POSTGRES_PASSWORD: admin
    restart: always
    ports:
      - '5432:5432'
    expose:
      - '5432'
    healthcheck:
      test: "exit 0"

  pgadmin-keycloak:
    image: dpage/pgadmin4
    container_name: pgadmin-keycloak
    restart: always
    ports:
      - "44:80"
    environment:
      PGADMIN_DEFAULT_EMAIL:
admin@gmail.com
      PGADMIN_DEFAULT_PASSWORD: admin
```

```
    volumes:
      -
pgadmin_keycloak_data_ex:/var/lib/pgadmin

  keycloak-service:
    image:
quay.io/keycloak/keycloak:latest
    container_name: keycloak-service
    environment:
      KC_DB: postgres
      KC_DB_URL:
jdbc:postgresql://postgres-keycloak-db:54
32/keycloak_db
      KC_DB_USERNAME: admin
      KC_DB_PASSWORD: admin
      KEYCLOAK_ADMIN: admin
      KEYCLOAK_ADMIN_PASSWORD: admin
      KC_HTTP_ENABLED: "true"
      KC_HOSTNAME_STRICT_HTTPS: "false"
    command:
      - start-dev
    restart: always
    ports:
      - '8080:8080'
    expose:
      - '8080'
    depends_on:
```

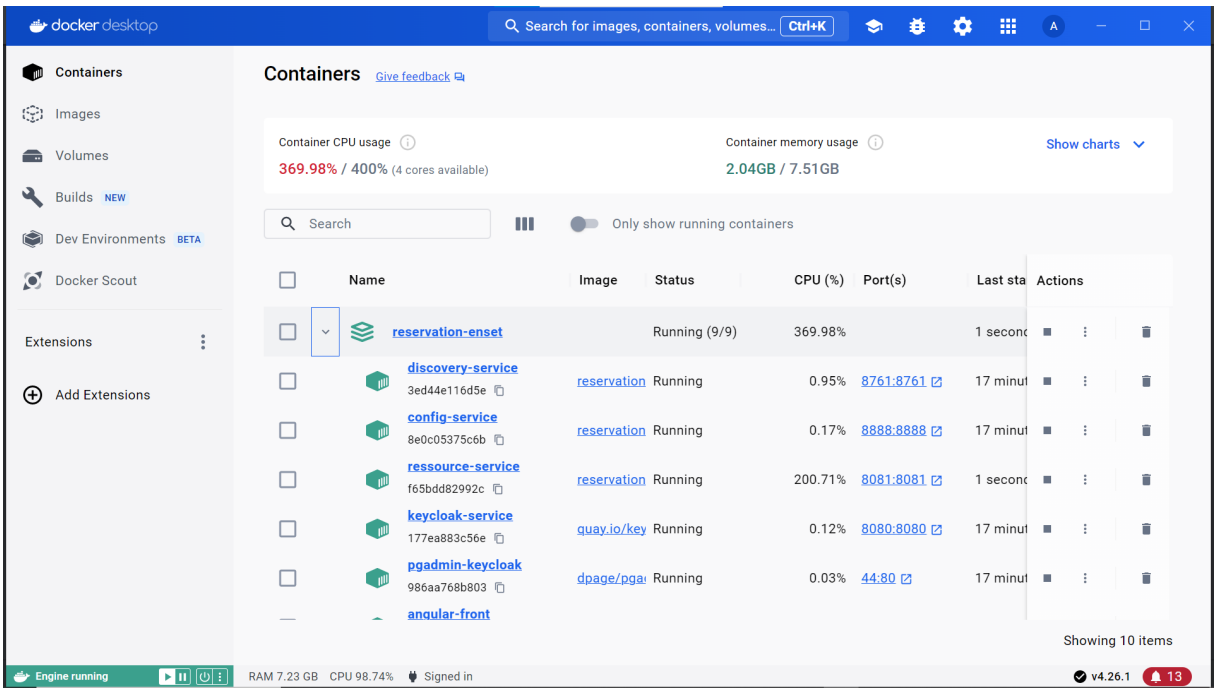
```
      - postgres-keycloak-db

  ressource-service:
    build: ./ressource-service
    container_name: ressource-service
    restart: always
    ports:
      - '8081:8081'
    expose:
      - '8081'
    environment:
      -
DISCOVERY_SERVICE_URL=http://discovery-se
rvice:8761/eureka
      -
CONFIG_SERVER=http://confi-service:9999
      -
ISSUER_URI=http://localhost:8080/realms/s
dia-realm
      -
JWK_SET_URI=http://keycloak-service:8080/
realms/sdia-realm/protocol/openid-connect
/certs
    depends_on:
      - config-service
      - keycloak-service

  reservation-service:
```

```
    build: ./reservation-service
    container_name: reservation-service
    restart: always
    ports:
      - '8082:8082'
    expose:
      - '8082'
    environment:
      -
DISCOVERY_SERVICE_URL=http://discovery-se
rvice:8761/eureka
      -
CONFIG_SERVER=http://config-service:9999
      -
ISSUER_URI=http://localhost:8080/realms/s
dia-realm
      -
JWK_SET_URI=http://keycloak-service:8080/
realms/sdia-realm/protocol/openid-connect
/certs
    depends_on:
      - ressource-service

  angular-front:
    build: ./angular-front-app
    container_name: angular-front
    restart: always
    ports:
```

```yaml
      - '8083:80'
    expose:
      - '8083'


volumes:
 pgadmin_keycloak_data_ex:
 postgres_keycloak_data_ex:
```

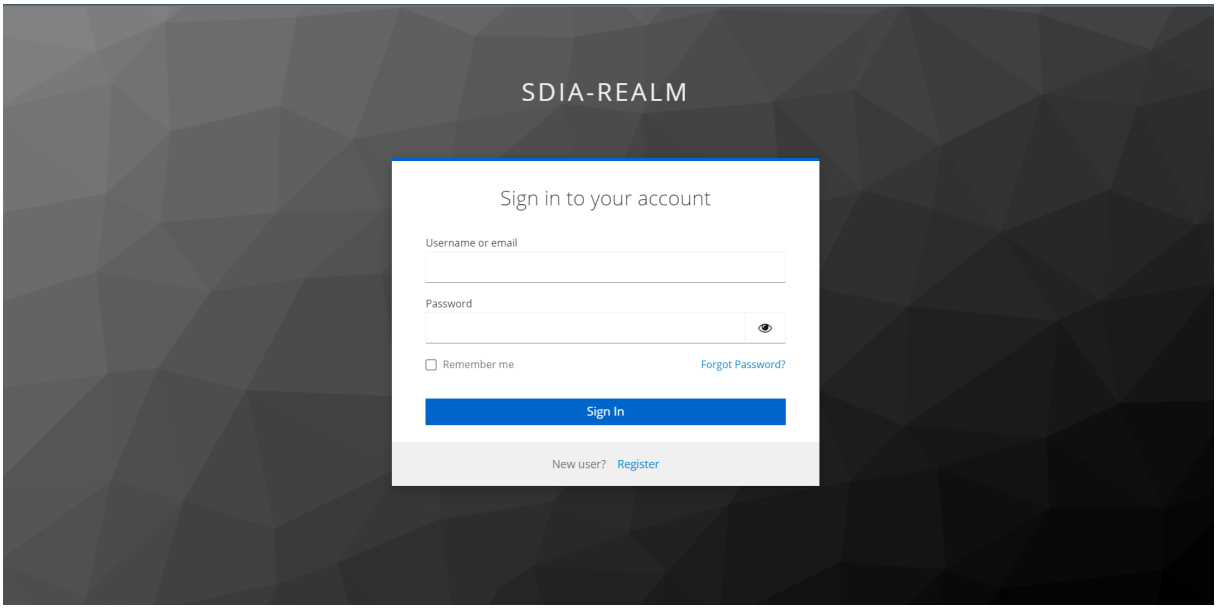
```
[+] Running 9/10covery-service          Started
 - Network reservation-enset_default    Created
 - Container postgres-keycloak-db       Started
 - Container angular-front              Started
 - Container discovery-service          Started
 - Container pgadmin-keycloak           Started
 - Container config-service             Started
 - Container keycloak-service           Started
 - Container gateway-service            Started
 - Container ressource-service          Started
 - Container reservation-service        Started
```

```
PS C:\Users\lenovo\IdeaProjects\reservation-enset> docker ps
CONTAINER ID   IMAGE                                    COMMAND                  CREATED          STATUS                    PORTS
          NAMES
467b6c149095   reservation-enset-reservation-service   "java -jar app.jar"      13 minutes ago   Up 22 seconds             0.0.0.0:8082->8082/tcp
          reservation-service
5c50529c257e   reservation-enset-gateway-service       "java -jar app.jar"      13 minutes ago   Up 13 minutes             0.0.0.0:9999->9999/tcp
          gateway-service
f65bdd82992c   reservation-enset-ressource-service     "java -jar app.jar"      13 minutes ago   Up 7 seconds              0.0.0.0:8081->8081/tcp
          ressource-service
177ea883c56e   quay.io/keycloak/keycloak:latest        "/opt/keycloak/bin/k…"   13 minutes ago   Up 13 minutes             0.0.0.0:8080->8080/tcp, 8
443/tcp   keycloak-service
8e0c05375c6b   reservation-enset-config-service        "java -jar app.jar"      13 minutes ago   Up 13 minutes             0.0.0.0:8888->8888/tcp
          config-service
986aa768b803   dpage/pgadmin4                          "/entrypoint.sh"         13 minutes ago   Up 13 minutes             443/tcp, 0.0.0.0:44->80/t
cp        pgadmin-keycloak
64fc365e4f70   reservation-enset-angular-front         "/docker-entrypoint.…"   13 minutes ago   Up 13 minutes             8083/tcp, 0.0.0.0:8083->8
0/tcp     angular-front
08346ba3d79d   postgres                                "docker-entrypoint.s…"   13 minutes ago   Up 13 minutes (healthy)   0.0.0.0:5432->5432/tcp
          postgres-keycloak-db
3ed44e116d5e   reservation-enset-discovery-service     "java -jar app.jar"      13 minutes ago   Up 13 minutes             0.0.0.0:8761->8761/tcp
          discovery-service
```

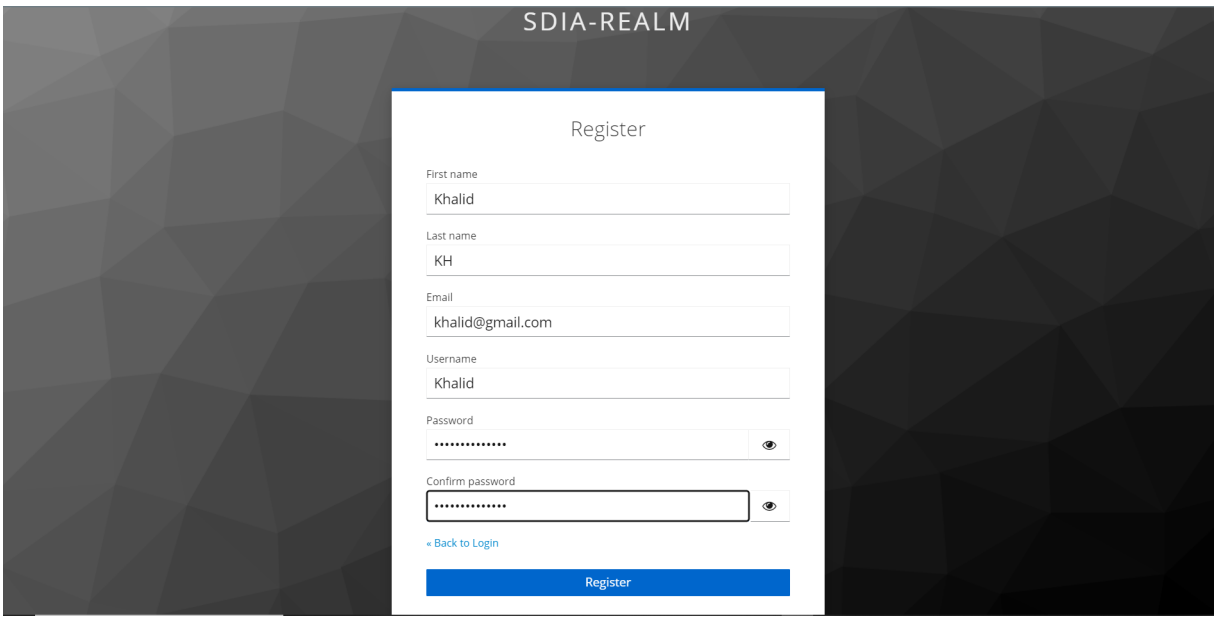


11. Tester l'Application:

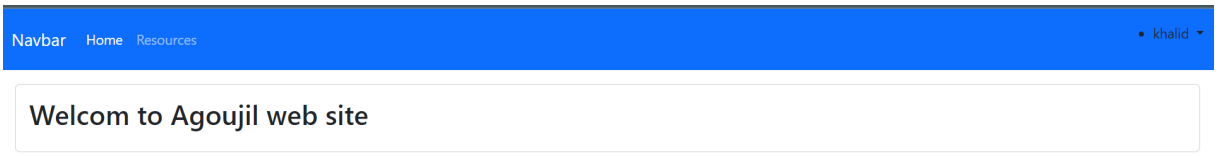
1. Authentification:



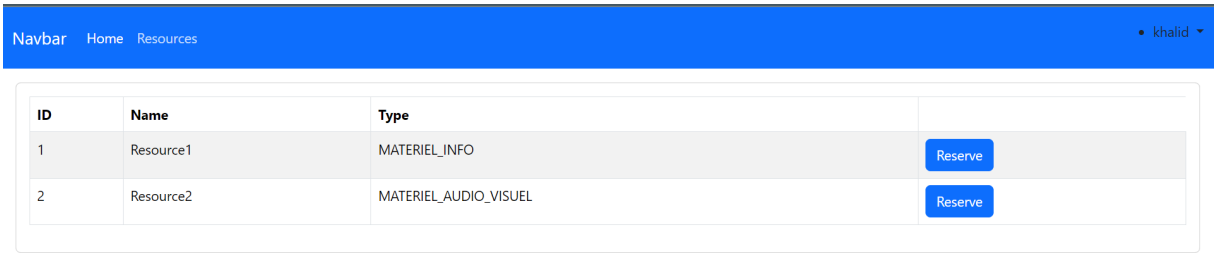
2. Register



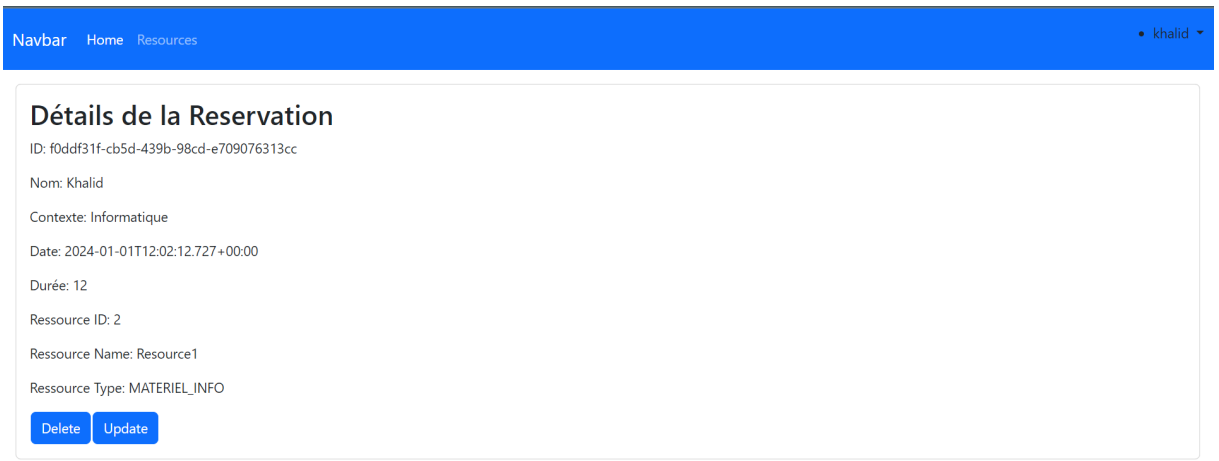
3. l'interface pour les clients

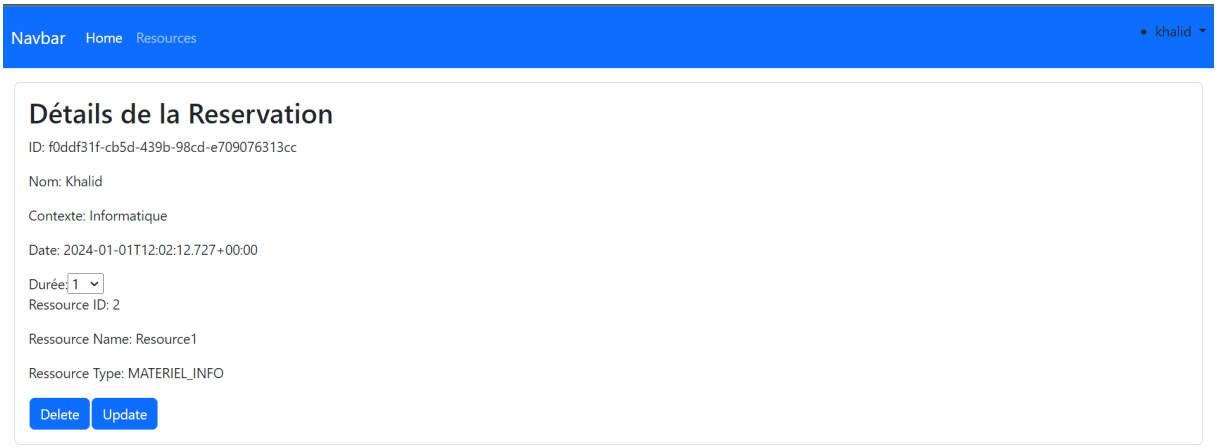


4. Réserver une ressource

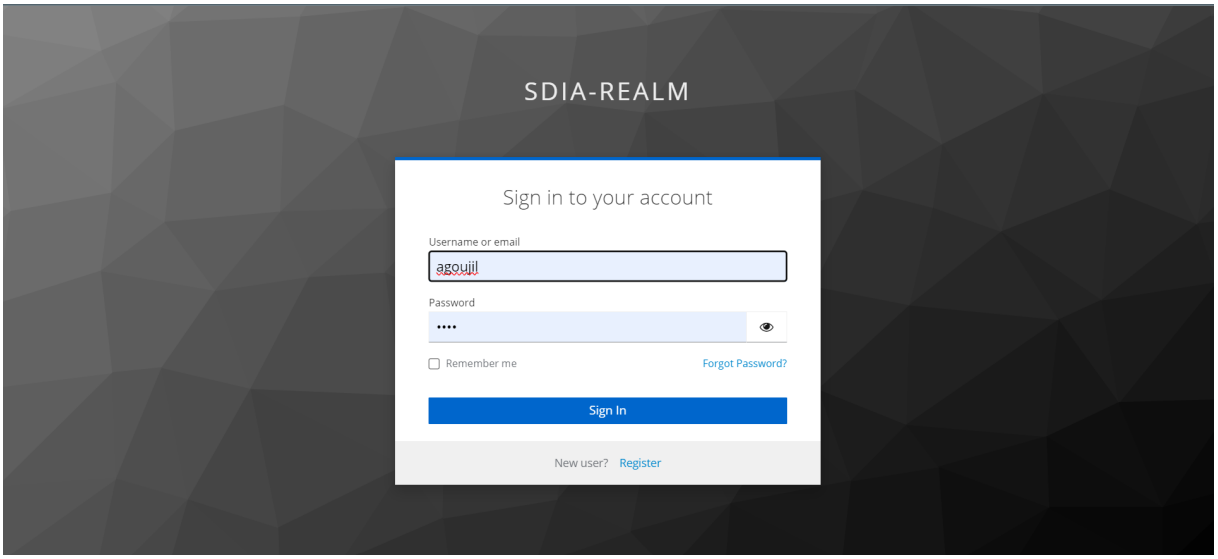


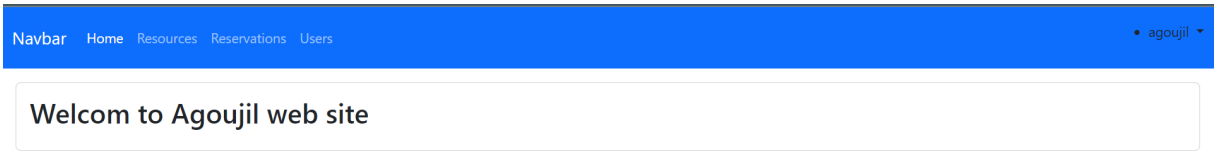
5. Consulter et modifier la réservation





6. l'interface pour l'Admin





7. Consulter les ressources

Navbar Home Resources Reservations Users • agoujil

ID	Name	Type	
1	Resource1	MATERIEL_INFO	Reserve
2	Resource2	MATERIEL_AUDIO_VISUEL	Reserve

8. Consulter les réservations

Navbar Home Resources Reservations Users • agoujil

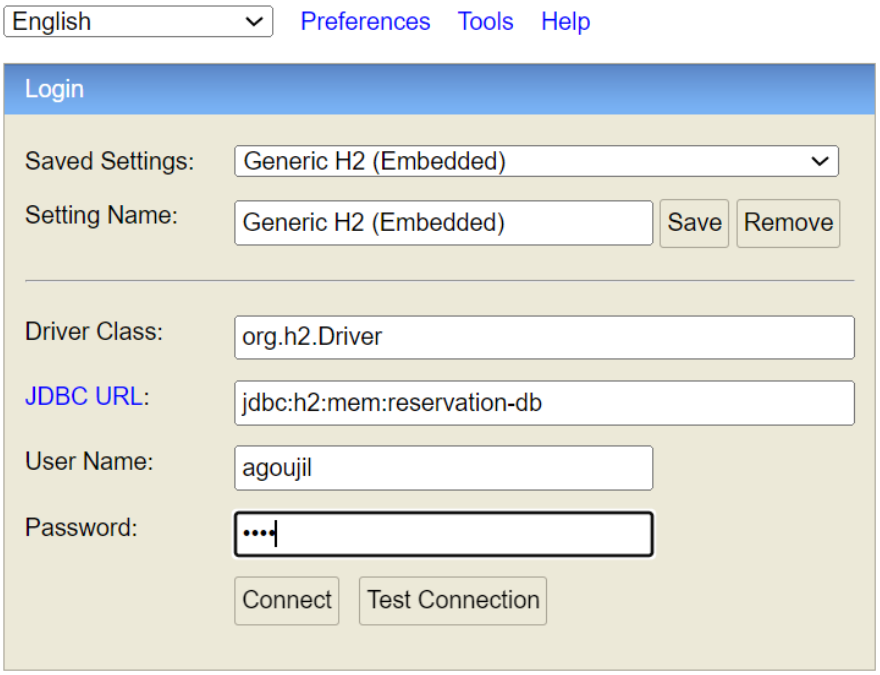
ID	Name	Context	Date	Duration	Ressource Id	Personne Id	
1	reservation1	context1	2024-01-01T14:40:24.126+00:00	16	1	0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0	Delete
2	reservation2	context2	2024-01-01T14:40:24.126+00:00	26	2	0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0	Delete
3	reservation1	context1	2024-01-01T14:40:24.153+00:00	24	1	43fb0c9e-290e-4c83-b320-0ae1367fd4a7	Delete
4	reservation2	context2	2024-01-01T14:40:24.153+00:00	24	2	43fb0c9e-290e-4c83-b320-0ae1367fd4a7	Delete
5	reservation70	Informatique	2024-01-01T14:41:40.463+00:00	5	1	6d6a390a-c02b-4c73-b819-17b062851f8f	Delete

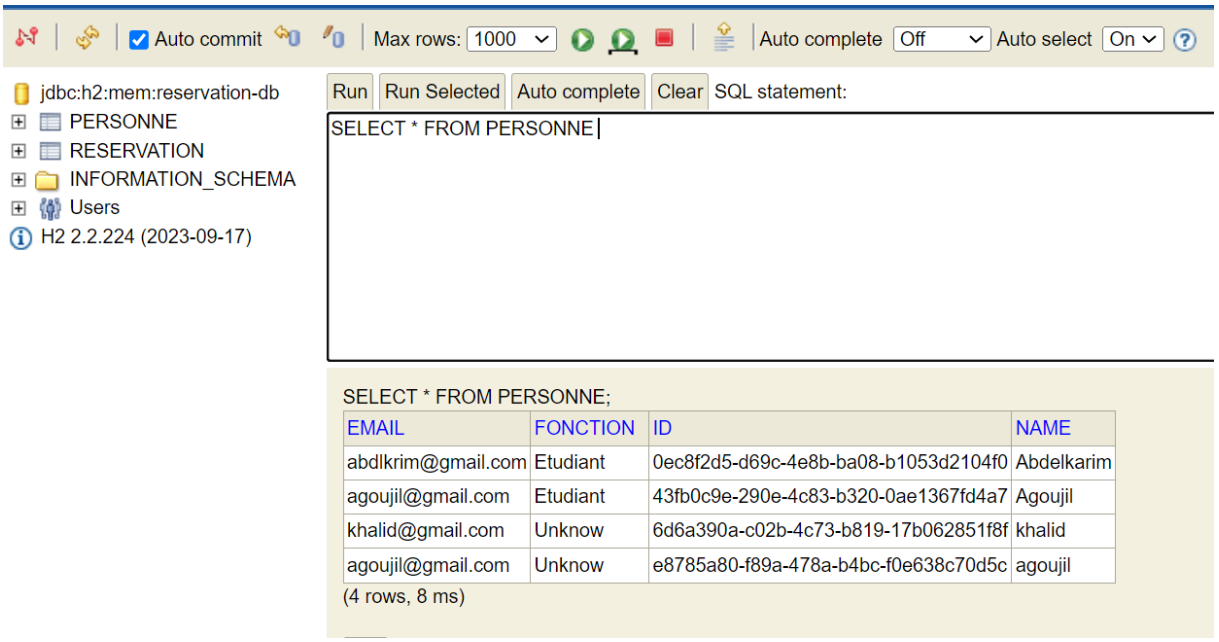
9. Consulter les utilisateurs

Navbar Home Resources Reservations Users • agoujil

ID	Name	Email	Fonction		
0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0	Abdelkarim	abdlkrim@gmail.com	Etudiant	Delete	Update
43fb0c9e-290e-4c83-b320-0ae1367fd4a7	Agoujil	agoujil@gmail.com	Etudiant	Delete	Update
6d6a390a-c02b-4c73-b819-17b062851f8f	khalid	khalid@gmail.com	Unknow	Delete	Update
e8785a80-f89a-478a-b4bc-f0e638c70d5c	agoujil	agoujil@gmail.com	Unknow	Delete	Update

10. Consulter H2 data base


English
Preferences
Tools
Help
Login
Saved Settings:
Generic H2 (Embedded)
Setting Name:
Generic H2 (Embedded)
Save
Remove
Driver Class:
org.h2.Driver
JDBC URL:
jdbc:h2:mem:reservation-db
User Name:
agoujil
Password:
Connect
Test Connection


Auto commit
Max rows: 1000
Auto complete Off
Auto select On
jdbc:h2:mem:reservation-db
PERSONNE
RESERVATION
INFORMATION_SCHEMA
Users
H2 2.2.224 (2023-09-17)
Run
Run Selected
Auto complete
Clear
SQL statement:
SELECT * FROM PERSONNE
SELECT * FROM PERSONNE;
EMAIL | FONCTION | ID | NAME
abdlkrim@gmail.com | Etudiant | 0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0 | Abdelkarim
agoujil@gmail.com | Etudiant | 43fb0c9e-290e-4c83-b320-0ae1367fd4a7 | Agoujil
khalid@gmail.com | Unknow | 6d6a390a-c02b-4c73-b819-17b062851f8f | khalid
agoujil@gmail.com | Unknow | e8785a80-f89a-478a-b4bc-f0e638c70d5c | agoujil
(4 rows, 8 ms)

Auto commit | Max rows: 1000 | Auto complete Off | Auto select On

jdbc:h2:mem:reservation-db
- PERSONNE
- RESERVATION
- INFORMATION_SCHEMA
- Users

H2 2.2.224 (2023-09-17)

Run | Run Selected | Auto complete | Clear | SQL statement:

```
SELECT * FROM RESERVATION
```

SELECT * FROM RESERVATION;

DURATION	DATE	ID	RESSOURCE_ID	CONTEXT	NAME	PERSONNE_ID
16	2024-01-01 15:40:24.126	1	1	context1	reservation1	0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0
26	2024-01-01 15:40:24.126	2	2	context2	reservation2	0ec8f2d5-d69c-4e8b-ba08-b1053d2104f0
24	2024-01-01 15:40:24.153	3	1	context1	reservation1	43fb0c9e-290e-4c83-b320-0ae1367fd4a7
24	2024-01-01 15:40:24.153	4	2	context2	reservation2	43fb0c9e-290e-4c83-b320-0ae1367fd4a7
5	2024-01-01 15:41:40.463	5	1	Informatique	reservation70	6d6a390a-c02b-4c73-b819-17b062851f8f

(5 rows, 6 ms)