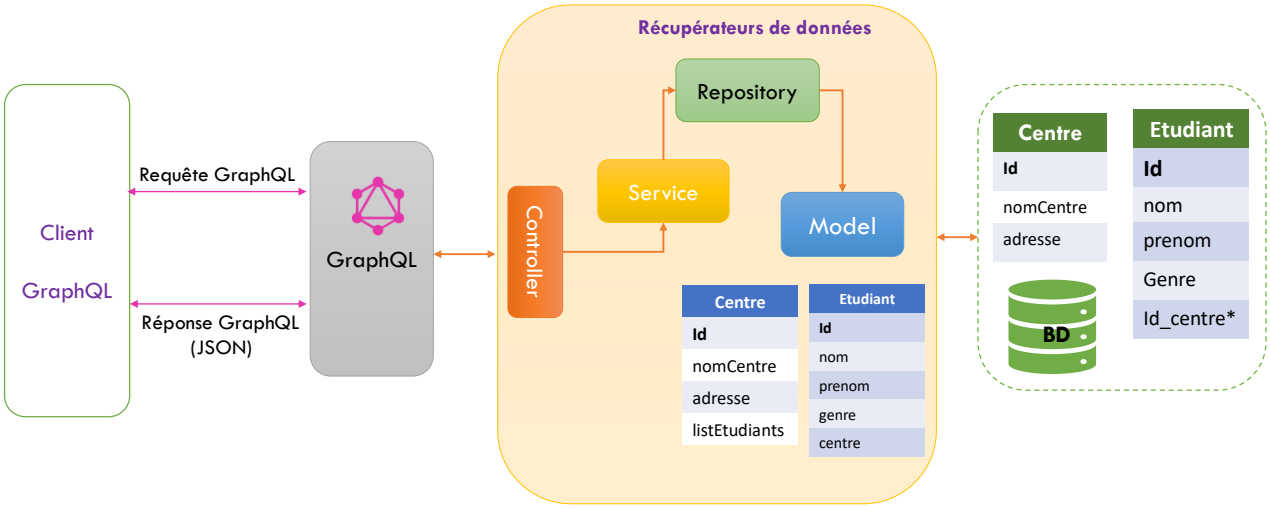
APPLICATION - SPRING DATA REST- GRAPHQL



APPLICATION - SPRING DATA REST- GRAPHQL

ÉTAPES À SUIVRE

i. Création d'un projet Spring Boot y compris les dépendances

ii. Ajout des Entités JPA

iii. Création des Repository JPA

iv. Ajout d'une classe Controller

v. Création des objets de transfert de données (DTO)

vi. Définition du schéma GraphQL

vii. Définition de la source de données

viii. Teste

- **Spring Web**
- **H2 Database**
- **Lombok**
- **Spring Data JPA**
- **Rest Repositories**
- **Spring for GraphQL**
- **WebSocket**

APPLICATION - SPRING DATA REST- GRAPHQL

Entités JPA

```java
@Entity
@Data
@Builder
@AllArgsConstructor
@NoArgsConstructor
public class Centre {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    Long id;
    String nom;
    String adresse;
    @OneToMany(mappedBy = "centre", cascade = CascadeType.ALL)
    List<Etudiant> listEtudiants;
}
```

```java
@Entity @Data
@AllArgsConstructor @NoArgsConstructor
@Builder @Table(name="etudiants")
public class Etudiant {
    @Id
    @GeneratedValue(strategy = GenerationType.AU
    Long id;
    @Column(name="nom_etudiant", nullable=false)
    String nom;
    @Column(name="prenom_etudiant")
    String prenom;
    @Enumerated(EnumType.STRING)
    Genre genre;
    @ManyToOne
    @NotNull
    @JoinColumn(name="centre_id")
    Centre centre;
}
```


APPLICATION - SPRING DATA REST- GRAPHQL

Création des objets de transfert de données (DTO)

```java
public record EtudiantDTO (
        String nom,
        String prenom,
        Genre genre,
        Long centreId
){   }
```

NB : Ajouter la classe CentreDTO

APPLICATION - SPRING DATA REST- GRAPHQL

Ajouter la classe de mapping : EtudiantDTO->Etudiant

```
@Component
public class DtoToEtudiant {
@Autowired
CentreRepository centreRepository;
public  void toEtudiant(Etudiant et, EtudiantDTO dto) {
    Centre centre=
centreRepository.findById(dto.centreId()).orElse(null);
    if (dto != null) {
        et.setNom(dto.nom());
        et.setPrenom(dto.prenom());
        et.setGenre(dto.genre());
        et.setCentre(centre);
    }
}

}
```


APPLICATION - SPRING DATA REST- GRAPHQL

3- Repository JPA

```
public interface CentreRepository extends JpaRepository<Centre,Long> {
}


public interface EtudiantRepository extends JpaRepository<Etudiant,Long>
{
}
```

APPLICATION - SPRING DATA REST- GRAPHQL

Services

```
@Service
public class EtudiantService {
    @Autowired
    DtoToEtudiant dtoToEtudiant;
    @Autowired
    EtudiantRepository etudiantRepository;

    private final Sinks.Many<Etudiant> sink = Sinks.many().multicast().onBackpressureBuffer();
    public List<Etudiant> getStudents() {
        return etudiantRepository.findAll();
    }
public Etudiant getEtudiant(Long id){
        return etudiantRepository.findById(id).orElse(null);
   }
public Etudiant addEtudiant(EtudiantDTO etudiantDTO) {
        Etudiant etudiant=new Etudiant();
        dtoToEtudiant.toEtudiant(etudiant, etudiantDTO);
        etudiantRepository.save(etudiant);
        sink.tryEmitNext(etudiant);

        return etudiant;
    }
```

SERVICE (SUITE)

```
    public Etudiant updateEtudiant(Long id, EtudiantDTO etudiantDTO){
        if(etudiantRepository.findById(id).isPresent()){
            Etudiant etudiant=etudiantRepository.findById(id).get();
            dtoToEtudiant.toEtudiant(etudiant,etudiantDTO);
            return etudiantRepository.save(etudiant);
        }
            return null;
    }
     public Flux<Etudiant> getEtudiantAddedPublisher() {
        return sink.asFlux();
     }
     public String deleteEtudiant(Long id){
        if(etudiantRepository.findById(id).isPresent()){
            etudiantRepository.deleteById(id);
            return String.format("l'étudiant %s est bien supprimé !",id);
        }

    ,
}
```

CONTROLLER

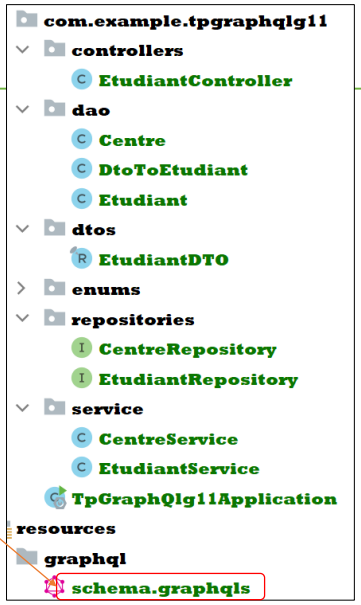
```
@Controller
public class EtudiantCentreController {
     @Autowired
    EtudiantService etudiantService;
     @Autowired
     CentreService centreService;
    @QueryMapping
    public List<Centre> getAllCentres(){
        return centreService.centres();
    }
    @QueryMapping
    public List<Etudiant>getAllEtudiants(){

        return etudiantService.getStudents();
    }
    @QueryMapping
    public Centre getCentre(@Argument int id){
        return centreService.getCentre(id);
    }
```


CONTROLLER(SUITE)

```
    @QueryMapping
    public Etudiant getEtudiant(@Argument Long id){
        return etudiantService.getEtudiant(id);
    }
    @MutationMapping
    public Etudiant addEtudiant(@Argument EtudiantDTO etudiantDTO) {
        return etudiantService.addEtudiant(etudiant);
    }
    @MutationMapping
    public String suppEtudiant(@Argument Long id){
        return etudiantService.deleteEtudiant(id);
    }
    @MutationMapping
    public Etudiant updateEtudiant(@Argument Long id,@Argument EtudiantDTO etudiantDTO){
        return etudiantService.updateEtudiant(id,etudiant);
    }
    @SubscriptionMapping
    public Flux<Etudiant> etudiantAdded() {
        return etudiantService.getEtudiantAddedPublisher();
    }
}
```

APPLICATION - SPRING DATA REST- GRAPHQL

Ajout d'un fichier graphqls : schema.graphqls



APPLICATION - SPRING DATA REST- GRAPHQL

```
type Query{
    listEtudiants : [Etudiant]
    getEtudiantById(id:Float):Etudiant
    centres:[Centre]
    getCentreById(id:Float):Centre
}
type Mutation{
    addEtudiant(etudiantDTO : EtudiantDTO):Etudiant
    updateEtudiant(id:Float,etudiantDTO : EtudiantDTO):Etudiant
    deleteEtudiant(id:Float):String
}
type Subscription{
    etudiantAdded:Etudiant
}
enum Genre {
    Homme,
    Femme
}
```

```
type Etudiant{
    id:Float
    nom:String
    prenom:String
    genre:Genre
    centre:Centre
}
type Centre{
    id:Int
    nom: String
    adresse:String
    listEtudiants:[Etudiant]
}
input EtudiantDTO{
    nom:String
    prenom:String
    genre:String
    centreId:Float
}
```

APPLICATION - SPRING DATA REST- GRAPHQL

6- Activation

```
spring.h2.console.enabled=true
spring.datasource.username=12
spring.datasource.password=
spring.datasource.url=jdbc:h2:mem:centredb
spring.graphql.graphiql.enabled=true
spring.graphql.websocket.path=/graphql
```

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


APPLICATION - SPRING DATA REST- GRAPHQL

7- Ajout d'un jeu d'enregistrements

```
public class TpGraphQlApplication implements CommandLineRunner{
    @Autowired
    EtudiantRepository etudiantRepository;
    @Autowired
    CentreRepository centreRepository;
    public static void main(String[] args) {
        SpringApplication.run(TpGraphQlApplication.class, args);
    }
    @Override
    public void run(String... args) throws Exception {
        Centre centre1=Centre.builder()
                .nom("Maarif").adresse("Biranzarane").build();
        centreRepository.save(centre1);
        Centre centre2=Centre.builder()
                .nom("Oranges").adresse("Oulfa").build();
        centreRepository.save(centre2);
        Etudiant et1=Etudiant.builder()
                .nom("Adnani").prenom("Brahim").genre(Genre.Homme)
                .centre(centre1).build();
        etudiantRepository.save(et1);
        …
```

8- TEST-QUERY

http://localhost:8080/graphiql?path=/graphql

```
query
{
  getEtudiantById(id: 1) {
    nom
    prenom
    centre {
      nom
      adresse
    }
  }
}
```

```
{
  "data": {
    "getEtudiantById": {
      "nom": "Adnani",
      "prenom": "Brahim",
      "centre": {
        "nom": "Maarif",
        "adresse": "Biranzarane"
      }
    }
  }
}
```

```
query{
  listEtudiants{
    id
    nom
    prenom
    centre{
      nom
      adresse
    }
  }
}
```

```
{
  "data": {
    "listEtudiants": [
      {
        "id": 1,
        "nom": "Adnani",
        "prenom": "Brahim",
        "centre": {
          "nom": "Maarif",
          "adresse": "Biranzarane"
        }
      },
      {
        "id": 2,
        "nom": "Alami",
        "prenom": "Ilham",
        "centre": {
          "nom": "Maarif",
          "adresse": "Biranzarane"
        }
      },
      {
        "id": 3,
        "nom": "Fadli",
        "prenom": "Moad",
        "centre": {
          "nom": "Oranges",
          "adresse": "Oulfa"
        }
      },
      {
        "id": 4,
        "nom": "Hamdani",
        "prenom": "Adil",
        "centre": {
          "nom": "Oranges",
          "adresse": "Oulfa"
        }
      }
    ]
  }
}
```

```
query{
  getCentreById(id:1){
    nom
    listEtudiants{
      nom
      prenom
    }
  }
}
```

```
{
  "data": {
    "getCentreById": {
      "nom": "Maarif",
      "listEtudiants": [
        {
          "nom": "Adnani",
          "prenom": "Brahim"
        },
        {
          "nom": "Alami",
          "prenom": "Ilham"
        }
      ]
    }
  }
}
```

```
fragment champsEtudiant on Etudiant{
  nom
  prenom
  genre
}
query{
  listEtudiants{
    id
    ...champsEtudiant
    centre{
      id
      nom
    }
  }
}
```


INSOMNIA POR GRAPHQL

POST http://localhost:8080/graphql Send

Params Body Auth Headers 4 Scripts D

GraphQL

Operations schema

```
query{
getEtudiant(id:3){
      nom
      genre
      }
}
```

Preview

```
{
    "data": {
        "getEtudiant": {
            "nom": "Fadli",
            "genre": "Homme"
        }
    }
}
```

8- TEST-MUTATION

```
mutation{
  updateEtudiant(id:1, etudiant:{
    nom:"UnNom"
    prenom:"UnPrénom"
    genre:"Homme"
    centreId:2
  })
  {
    nom
    prenom
    centre{
      id
      nom
    }
  }
}
```

```
{
  "data": {
    "updateEtudiant": {
      "nom": "UnNom",
      "prenom": "UnPrénom",
      "centre": {
        "id": 2,
        "nom": "Oranges"
      }
    }
  }
}
```

```
mutation{
  deleteEtudiant(id:5)
}
```

```
{
  "data": {
    "deleteEtudiant": "L'étudiant 5 bien supprimé "
  }
}
```

```
mutation{
  deleteEtudiant(id:10)
}
```

```
{
  "data": {
    "deleteEtudiant": "L'étudiant 10 n'exite pas "
  }
}
```

```
mutation($n:String,$p:String,$g:String,$cId:Float){
addEtudiant(etudiant: {
    nom:$n
    prenom:$p
    genre:$g
   centreId:$cId
 })
   {
      id
      nom
      centre{
        id
      }
    }
}
```

Variables Headers

```
{"n": "unNom","p":"unPrénom","g": "Homme","cId": 2 }
```


8- TEST-MUTATION (SUITE)

```
mutation($d:Int,$n:String,$p:String,$g:String,$i:Int){
  updateEtudiant(id:$d,etudiant:{
    nom:$n
    prenom:$p
    genre:$g
    centreId:$i
  }){
    nom
    prenom
    centre
    {
```

Variables Headers

```
{"d":4,"n":"testNom3","p":"testPrenom3","g":"Homme","i":2}
```

```
{
  "data": {
    "updateEtudiant": {
      "nom": "testNom3",
      "prenom": "testPrenom3",
      "centre": {
        "id": 2,
        "nom": "Oranges"
      }
    }
  }
}
```

```
mutation($et:EtudiantDTO){
  updateEtudiant(id:3,etudiant:$et)
   {
    nom
    prenom
    centre
    {
      id
      nom
    }
  }
}
```

Variables Headers

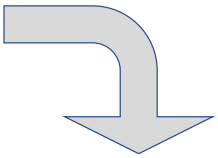
```
{"et":{"nom":"testNom30","prenom":"testPrenom30","genre":"Homme","centreId":2}}
```

```
{
  "data": {
    "updateEtudiant": {
      "nom": "testNom30",
      "prenom": "testPrenom30",
      "centre": {
        "id": 2,
        "nom": "Oranges"
      }
    }
  }
}
```

8- TEST-SUBSCRIPTION

```
mutation{
  addEtudiant(etudiant:{
    nom:"nom5"
    prenom:"prénom5"
    genre:"Homme"
    centreId:1
  }){
    id
    nom
    prenom
  }
}
```

```
{
  "data": {
    "addEtudiant": {
      "id": 5,
      "nom": "nom5",
      "prenom": "prénom5"
    }
  }
}
```



Client1

```
subscription{
  etudiantAdded{
    id
    nom
    prenom
    genre
    centre
    {
      id
      nom
      adresse
    }
  }
}
```

```
{
  "data": {
    "etudiantAdded": {
      "id": 5,
      "nom": "nom5",
      "prenom": "prénom5",
      "genre": "Homme",
      "centre": {
        "id": 1,
        "nom": "Maarif",
        "adresse": "Biranzarane"
      }
    }
  }
}
```

Client2

Architecture des composants