

# Examen Architecture JEE et Middlewares

---

Réalisé par :

AKASMIOU Ouassima

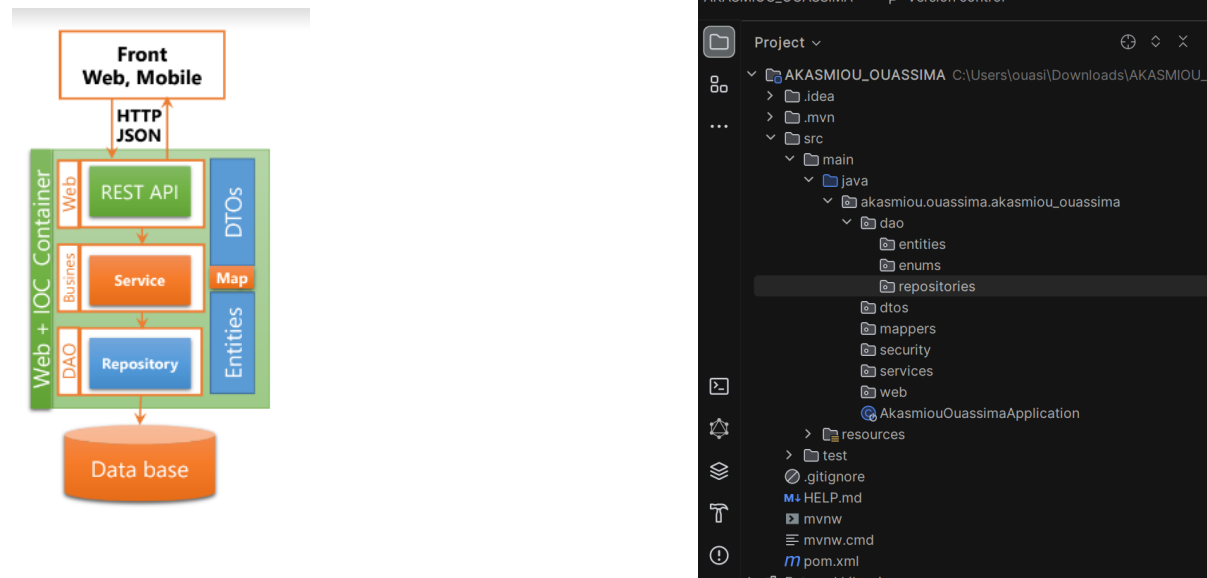
Glsid 2

Professeur :

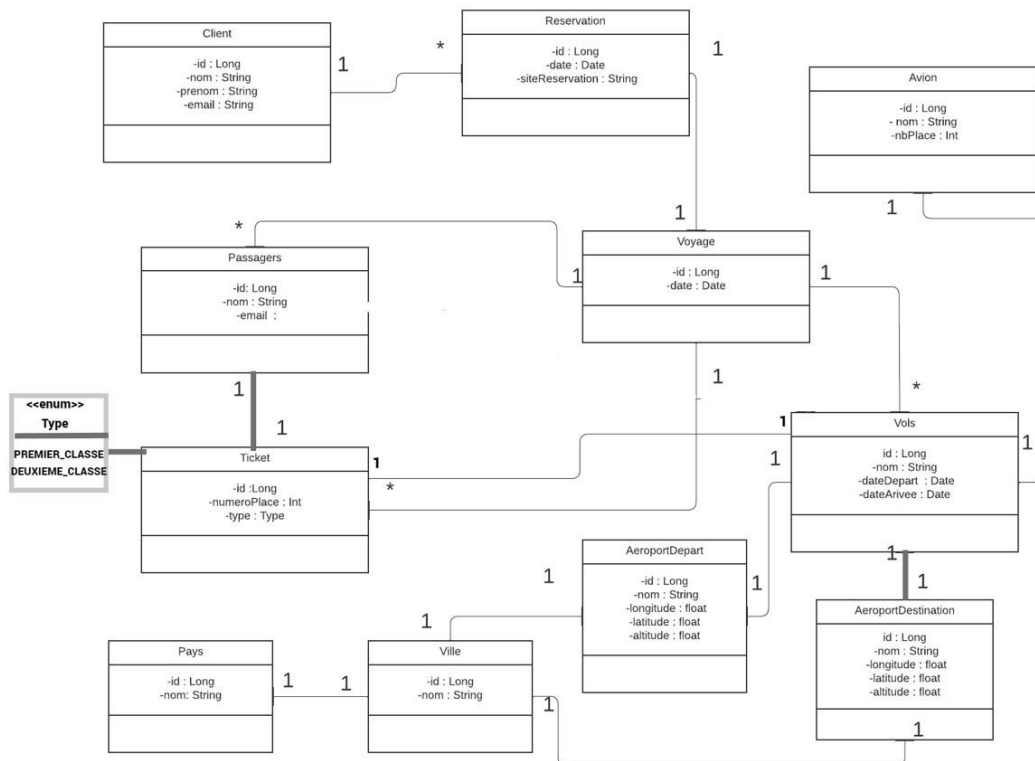
Mr Mohamed Youssfi

## A. Conception :

### 1. Établir une architecture technique du projet

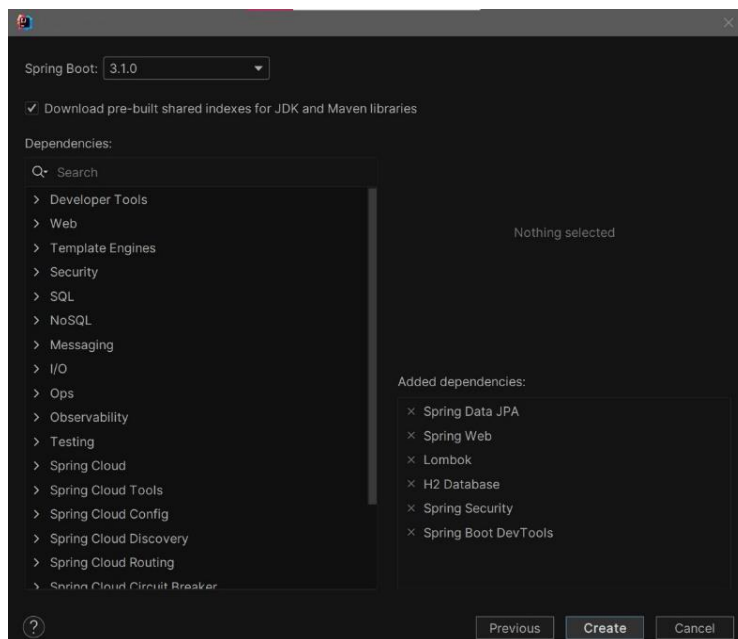
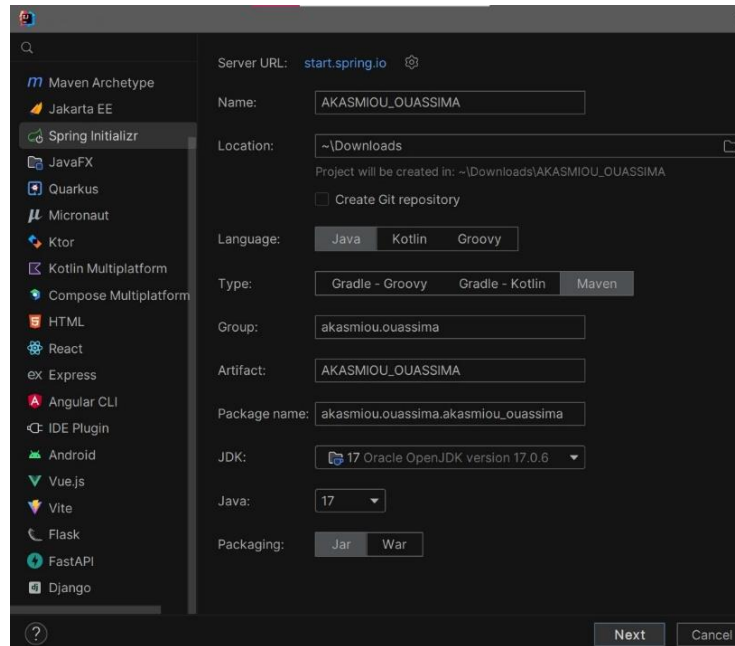


### 2. Établir un diagramme de classes qui montre les entités. On ne représentera que les attributs.



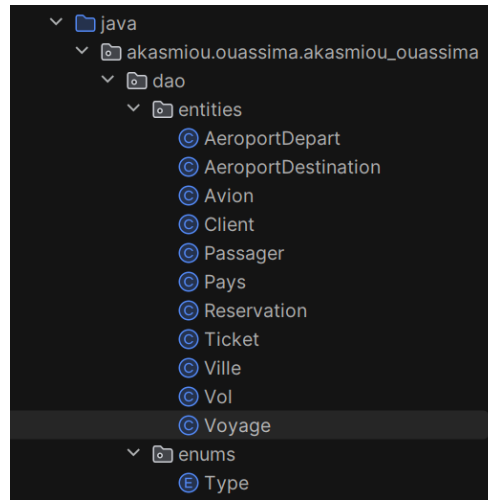
## B. Implémentation :

1. Créer un projet Spring boot avec les dépendances requises. Les identifiants du projet **GroupId**, **ArtifactId** et le package de base doivent contenir votre nom et prénom.



## 2. Couche DAO

### a. Créer les entités JPA



Voyage :

```
package akasmiau.ouassima.akasmiau_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.format.annotation.DateTimeFormat;

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

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Voyage {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Temporal(TemporalType.DATE)
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date date;
    @OneToMany(mappedBy = "voyage")
    private List<Passager> passagers;
    @OneToOne
    private Reservation reservation;
    @OneToMany(mappedBy = "voyage")
    private List<Ticket> tickets;
    @OneToMany(mappedBy = "voyage")
    private List<Vol> vols;
}
```

## Vol :

```
package akasmiau.ouassima.akasmiau_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.format.annotation.DateTimeFormat;
import java.util.Date;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Vol {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    @Temporal(TemporalType.DATE)
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date dateDepart;
    @Temporal(TemporalType.DATE)
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date dateArrivee;

    @OneToOne(mappedBy = "vol" )
    private AeroportDepart aeroportDepart;
    @OneToOne(mappedBy = "vol" )
    private AeroportDestination aeroportDestination;
    @OneToOne
    private Avion avion;
    @OneToOne
    private Ticket ticket;
    @ManyToOne(fetch = FetchType.LAZY)
    private Voyage voyage;
}
```

## Ville :

```
package akasmiau.ouassima.akasmiau_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Ville {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    @OneToOne(mappedBy = "ville" )
```

```

private AeroportDepart aeroportDepart;
@OneToOne(mappedBy = "ville" )
private AeroportDestination aeroportDestination;
@OneToOne(mappedBy = "ville" )
private Pays pays;
}

```

#### Ticket :

```

package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import akasmiou.ouassima.akasmiou_ouassima.dao.enums.Type;
import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Ticket {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private int numeroPlace;
    @Enumerated(EnumType.STRING)
    private Type type;
    @OneToOne
    private Passager passager;
    @OneToOne(mappedBy = "ticket")
    private Vol vol;
    @ManyToOne(fetch = FetchType.LAZY)
    private Voyage voyage;
}

```

#### Enum Type :

```

package akasmiou.ouassima.akasmiou_ouassima.dao.enums;

public enum Type {
    PREMIERE_CLASSE, DEUXIEME_CLASSE
}

```

#### Reservation :

```

package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.format.annotation.DateTimeFormat;
import java.util.Date;

@Entity
@Data
@AllArgsConstructor

```

```

@NoArgsConstructor
public class Reservation {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Temporal(TemporalType.DATE)
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date date;
    private String siteReservation;
    @ManyToOne(fetch = FetchType.LAZY)
    private Client client;
    @OneToOne(mappedBy = "reservation")
    private Voyage voyage;
}

```

### Pays :

```

package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Pays {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    @OneToOne()
    private Ville ville;
}

```

### Passager :

```

package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Passager {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    private String email;
    @ManyToOne
    private Voyage voyage;
    @OneToOne(mappedBy = "passager")
}

```

```
    private Ticket ticket;
}
```

#### Client :

```
package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import java.util.List;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Client {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    private String prenom;
    private String email;
    @OneToMany(mappedBy = "client")
    private List<Reservation> reservations;
}
```

#### Avion :

```
package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Avion {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    private int nbPlace;
    @OneToOne(mappedBy = "avion")
    private Vol vol;
}
```



## AéroportDestination :

```
package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class AéroportDestination {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    private float longitude;
    private float latitude;
    private float altitude;
    @OneToOne
    private Vol vol;
    @OneToOne
    private Ville ville;
}
```

## AéroportDepart :

```
package akasmiou.ouassima.akasmiou_ouassima.dao.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class AéroportDepart {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    private float longitude;
    private float latitude;
    private float altitude;
    @OneToOne
    private Vol vol;
    @OneToOne
    private Ville ville;
}
```

## b. Créer les interfaces JPA Repository basées sur Spring Data

```
package akasmiou.ouassima.akasmiou_ouassima.dao.repositories;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.AeroportDepart;
import org.springframework.data.jpa.repository.JpaRepository;

public interface AeroportDepartRepository extends
JpaRepository<AeroportDepart, Long> {
}
```

```
package akasmiou.ouassima.akasmiou_ouassima.dao.repositories;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.AeroportDestination;
import org.springframework.data.jpa.repository.JpaRepository;

public interface AeroportDestinationRepository extends
JpaRepository<AeroportDestination, Long> {
}
```

```
package akasmiou.ouassima.akasmiou_ouassima.dao.repositories;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.Avion;
import org.springframework.data.jpa.repository.JpaRepository;

public interface AvionRepository extends JpaRepository<Avion, Long> {
}
```

```
package akasmiou.ouassima.akasmiou_ouassima.dao.repositories;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.Client;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ClientRepository extends JpaRepository<Client, Long> {
}
```

```
package akasmiou.ouassima.akasmiou_ouassima.dao.repositories;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.Passager;
import org.springframework.data.jpa.repository.JpaRepository;

public interface PassagerRepository extends JpaRepository<Passager, Long> {
}
```

```
package akasmiou.ouassima.akasmiou_ouassima.dao.repositories;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.Pays;
import org.springframework.data.jpa.repository.JpaRepository;

public interface PaysRepository extends JpaRepository<Pays, Long> {
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dao.repositories;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Reservation;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ReservationRepository extends JpaRepository<Reservation,
Long> {
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dao.repositories;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Ticket;
import org.springframework.data.jpa.repository.JpaRepository;

public interface TicketRepository extends JpaRepository<Ticket, Long> {
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dao.repositories;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Ville;
import org.springframework.data.jpa.repository.JpaRepository;

public interface VilleRepository extends JpaRepository<Ville, Long> {
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dao.repositories;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Vol;
import org.springframework.data.jpa.repository.JpaRepository;

public interface VolRepository extends JpaRepository<Vol, Long> {
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dao.repositories;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Voyage;
import org.springframework.data.jpa.repository.JpaRepository;

public interface VoyageRepository extends JpaRepository<Voyage, Long> {
}
```

**c. Tester la couche DAO avec une application qui alimente la base de données avec quelques enregistrements de test.**

```
package akasmiou.ouassima.akasmiou_ouassima;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.*;
import akasmiou.ouassima.akasmiou_ouassima.dao.repositories.*;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.stream.Stream;

@SpringBootApplication
public class AkasmiouOuassimaApplication {

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

    @Bean
    CommandLineRunner start(AeroportDepartRepository
aeroportDepartRepository,
                           AeroportDestinationRepository
aeroportDestinationRepository,
                           ClientRepository clientRepository,
                           ReservationRepository reservationRepository,
                           VolRepository volRepository,
                           VoyageRepository voyageRepository,
                           VilleRepository villeRepository,
                           PaysRepository paysRepository,
                           PassengerRepository passengerRepository,
                           TicketRepository ticketRepository,
                           AvionRepository avionRepository
    ){

        return args -> {
            List<Client> clients = new ArrayList<>();
            Stream.of("ouassima", "mohamed", "jinan").forEach(name ->{
                Client client = new Client();
                client.setNom(Math.random() < 0.5? "akasmiou": "khladi");
                client.setPrenom(name);
                client.setEmail(name+"@gmail.com");
                clientRepository.save(client);
                clients.add(client);
            });

            Reservation reservation1 = new Reservation();
            reservation1.setDate(new Date());
            reservation1.setSiteReservation("www.reservation.com");
            reservation1.setClient(clients.get(0));
            reservationRepository.save(reservation1);
            Reservation reservation2 = new Reservation();
            reservation2.setDate(new Date());
            reservation2.setSiteReservation("www.reservation.com");
```

```

reservation2.setClient(clients.get(1));
reservationRepository.save(reservation2);
Reservation reservation3 = new Reservation();
reservation3.setDate(new Date());
reservation3.setSiteReservation("www.reservation.com");
reservation3.setClient(clients.get(2));
reservationRepository.save(reservation3);

Voyage voyage1 = new Voyage();
voyage1.setDate(new Date());
voyage1.setReservation(reservation1);
voyageRepository.save(voyage1);

Voyage voyage2 = new Voyage();
voyage2.setDate(new Date());
voyage2.setReservation(reservation2);
voyageRepository.save(voyage2);

Voyage voyage3 = new Voyage();
voyage3.setDate(new Date());
voyage3.setReservation(reservation3);
voyageRepository.save(voyage3);

Vol vol1= new Vol();
vol1.setNom("vol1");
vol1.setDateDepart(new Date());
vol1.setDateArrivee(new Date());
vol1.setVoyage(voyage1);
volRepository.save(vol1);
Vol vol2= new Vol();
vol2.setNom("vol2");
vol2.setDateDepart(new Date());
vol2.setDateArrivee(new Date());
vol2.setVoyage(voyage2);
volRepository.save(vol2);
Vol vol3= new Vol();
vol3.setNom("vol3");
vol3.setDateDepart(new Date());
vol3.setDateArrivee(new Date());
vol3.setVoyage(voyage3);
volRepository.save(vol3);
Vol vol4= new Vol();
vol4.setNom("vol4");
vol4.setDateDepart(new Date());
vol4.setDateArrivee(new Date());
vol4.setVoyage(voyage3);
volRepository.save(vol4);

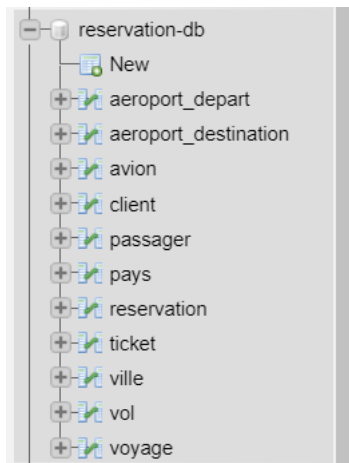
Avion avion1 = new Avion();
avion1.setNom("avion1");
avion1.setNbPlace(100);
avion1.setVol(vol1);
avionRepository.save(avion1);
Avion avion2 = new Avion();
avion2.setNom("avion2");
avion2.setNbPlace(300);
avion2.setVol(vol2);

```

```
        avionRepository.save(avion2);
        Avion avion3 = new Avion();
        avion3.setNom("avion3");
        avion3.setNbPlace(200);
        avion3.setVol(vol3);
        avionRepository.save(avion3);

    };

}
```



## Table Reservation

					date	client_id	id	site_reservation
<input type="checkbox"/>	Edit	Copy	Delete		2023-05-22		1	www.reservation.com
<input type="checkbox"/>	Edit	Copy	Delete		2023-05-22		2	www.reservation.com
<input type="checkbox"/>	Edit	Copy	Delete		2023-05-22		3	www.reservation.com

## Table Client

					id	email	nom	prenom
<input type="checkbox"/>	Edit	Copy	Delete		1	ouassima@gmail.com	khladi	ouassima
<input type="checkbox"/>	Edit	Copy	Delete		2	mohamed@gmail.com	khladi	mohamed
<input type="checkbox"/>	Edit	Copy	Delete		3	jinan@gmail.com	akasmiau	jinan



## Table Aeroport\_Depart

```
SELECT * FROM `aeroport_depart`
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

altitude	latitude	longitude	id	ville_id	vol_id	nom
----------	----------	-----------	----	----------	--------	-----

Query results operations

## Table Aeroport\_Destination

```
SELECT * FROM `aeroport_destination`
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

altitude	latitude	longitude	id	ville_id	vol_id	nom
----------	----------	-----------	----	----------	--------	-----

Query results operations

## Table Passager

```
SELECT * FROM `passager`
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

id	voyage_id	email	nom
----	-----------	-------	-----

Query results operations

## Table Pays

```
SELECT * FROM `pays`
```

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

id	ville_id	nom
----	----------	-----

Query results operations



## Table Ticket

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
SELECT * FROM `ticket`
```

☐ Profiling [ [Edit inline](#) ] [ [Edit](#) ] [ [Explain SQL](#) ] [ [Create PHP code](#) ] [ [Refresh](#) ]

numero_place	id	passager_id	voyage_id	type
--------------	----	-------------	-----------	------

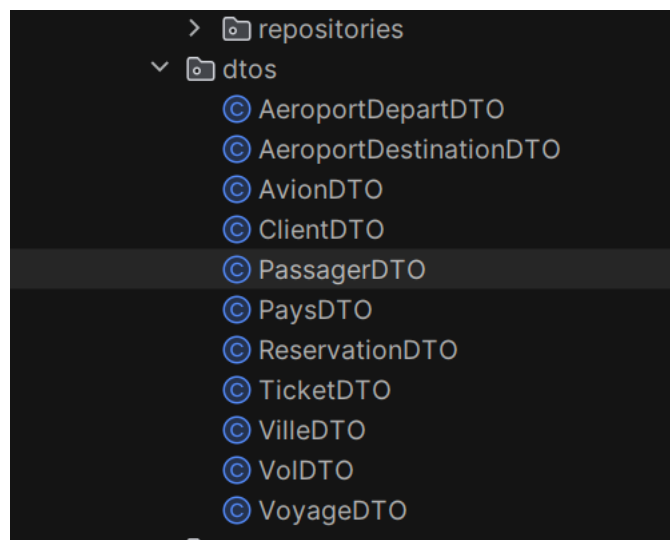
Query results operations

Create view

### 3. Couche Web REST API :

En créant les DTO et les mappeurs requis :

DTOS :



```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.*;
import jakarta.persistence.*;
import lombok.Data;
import org.springframework.format.annotation.DateTimeFormat;

import java.util.Date;

@Data
public class VolDTO {

    private Long id;
    private String nom;
```

```

    private Date dateDepart;

    private Date dateArrivee;
    private AvionDTO avionDTO;
    private VoyageDTO voyageDTO;
}

```

```

package akasmiau.ouassima.akasmiau_ouassima.dtos;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Client;
import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Voyage;
import jakarta.persistence.*;
import lombok.Data;
import org.springframework.format.annotation.DateTimeFormat;

import java.util.Date;
@Data
public class ReservationDTO {

    private Long id;

    private Date date;
    private String siteReservation;

    private ClientDTO clientDTO;

    private VoyageDTO voyageDTO;
}

```

```

package akasmiau.ouassima.akasmiau_ouassima.dtos;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Reservation;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.OneToMany;
import lombok.Data;

import java.util.List;
@Data
public class ClientDTO {

    private Long id;
    private String nom;
    private String prenom;
    private String email;
}

```

```

package akasmiau.ouassima.akasmiau_ouassima.dtos;

import lombok.Data;

```

```
@Data
public class PaysDTO {
    private Long id;
    private String nom;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import akasmiau.ouassima.akasmiau_ouassima.dao.enums.Type;
import jakarta.persistence.EnumType;
import jakarta.persistence.Enumerated;
import lombok.Data;

@Data
public class TicketDTO {
    private Long id;
    private int numeroPlace;

    private Type type;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import lombok.Data;

@Data
public class VilleDTO {
    private Long id;
    private String nom;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import jakarta.persistence.Temporal;
import jakarta.persistence.TemporalType;
import lombok.Data;
import org.springframework.format.annotation.DateTimeFormat;

import java.util.Date;
@Data
public class VoyageDTO {
    private Long id;

    private Date date;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import lombok.Data;

@Data
public class AvionDTO {
```

```
private Long id;
private String nom;
private int nbPlace;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import lombok.Data;

@Data
public class AeroportDepartDTO {
    private Long id;
    private String nom;
    private float longitude;
    private float latitude;
    private float altitude;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import lombok.Data;

@Data
public class AeroportDestinationDTO {
    private Long id;
    private String nom;
    private float longitude;
    private float latitude;
    private float altitude;
}
```

```
package akasmiau.ouassima.akasmiau_ouassima.dtos;

import lombok.Data;

@Data
public class PassagerDTO {
    private Long id;
    private String nom;
    private String email;
}
```

## Mappeurs :

```
package akasmiou.ouassima.akasmiou_ouassima.mappers;

import akasmiou.ouassima.akasmiou_ouassima.dao.entities.*;
import akasmiou.ouassima.akasmiou_ouassima.dtos.*;
import org.springframework.beans.BeanUtils;
import org.springframework.stereotype.Service;

@Service
public class ReservationMapperImpl {
    public VolDTO fromVol(Vol vol) {
        VolDTO volDTO = new VolDTO();
        BeanUtils.copyProperties(vol, volDTO);
        volDTO.setAvionDTO(fromAvion(vol.getAvion()));
        volDTO.setVoyageDTO(fromVoyage(vol.getVoyage()));
        return volDTO;
    }
    public Vol fromVolDTO(VolDTO volDTO) {
        Vol vol= new Vol();
        BeanUtils.copyProperties(volDTO, vol);
        return vol;
    }
    public ReservationDTO fromReservation(Reservation reservation) {
        ReservationDTO reservationDTO = new ReservationDTO();
        BeanUtils.copyProperties(reservation, reservationDTO);
        reservationDTO.setClientDTO(fromClient(reservation.getClient()));
        reservationDTO.setVoyageDTO(fromVoyage(reservation.getVoyage()));
        return reservationDTO;
    }
    public Reservation fromReservationDTO(ReservationDTO reservationDTO) {
        Reservation reservation = new Reservation();
        BeanUtils.copyProperties(reservationDTO, reservation);
        return reservation;
    }

    public AvionDTO fromAvion(Avion avion) {
        AvionDTO avionDTO = new AvionDTO();
        BeanUtils.copyProperties(avion, avionDTO);
        return avionDTO;
    }
    public Avion fromAvionDTO(AvionDTO avionDTO) {
        Avion avion= new Avion();
        BeanUtils.copyProperties(avionDTO, avion);
        return avion;
    }
    public VoyageDTO fromVoyage(Voyage voyage) {
        VoyageDTO voyageDTO = new VoyageDTO();
        BeanUtils.copyProperties(voyage, voyageDTO);
        return voyageDTO;
    }
    public Voyage fromVoyageDTO(VoyageDTO voyageDTO) {
        Voyage voyage = new Voyage();
        BeanUtils.copyProperties(voyageDTO, voyage);
        return voyage;
    }
    public ClientDTO fromClient(Client client) {
        ClientDTO clientDTO = new ClientDTO();
    }
}
```

```

        BeanUtils.copyProperties(client, clientDTO);
        return clientDTO;
    }
    public Client fromClientDTO(ClientDTO clientDTO) {
        Client client = new Client();
        BeanUtils.copyProperties(clientDTO, client);
        return client;
    }
}

```

```

package akasmiau.ouassima.akasmiau_ouassima.services;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Vol;
import akasmiau.ouassima.akasmiau_ouassima.dtos.ReservationDTO;
import akasmiau.ouassima.akasmiau_ouassima.dtos.VolDTO;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public interface ReservationService {
    VolDTO saveVol(VolDTO volDTO);
    List<VolDTO> listVols();
    VolDTO getVol(Long volId);
    VolDTO updateVol(VolDTO volDTO);
    void deleteVol(Long volId);
    ReservationDTO saveReservation(ReservationDTO reservationDTO);
    List<ReservationDTO> listReservations();
    ReservationDTO getReservation(Long reservationId);
    ReservationDTO updateReservation(ReservationDTO reservationDTO);
    void deleteReservation(Long reservationId);
}

```

```

package akasmiau.ouassima.akasmiau_ouassima.services;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Reservation;
import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Vol;
import akasmiau.ouassima.akasmiau_ouassima.dao.repositories.ReservationRepository;
import akasmiau.ouassima.akasmiau_ouassima.dao.repositories.VolRepository;
import akasmiau.ouassima.akasmiau_ouassima.dtos.ReservationDTO;
import akasmiau.ouassima.akasmiau_ouassima.dtos.VolDTO;
import akasmiau.ouassima.akasmiau_ouassima.mappers.ReservationMapperImpl;

import java.util.List;
import java.util.stream.Collectors;

@Service
@Transactional
@AllArgsConstructor
public class ReservationServiceImpl implements ReservationService {
    VolRepository volRepository;
    ReservationRepository reservationRepository;
    ReservationMapperImpl dtoMapper;
}

```

```

@Override
public VolDTO saveVol(VolDTO volDTO) {
    Vol vol = dtoMapper.fromVolDTO(volDTO);
    Vol savedVol = volRepository.save(vol);
    return dtoMapper.fromVol(savedVol);
}

@Override
public List<VolDTO> listVols() {
    List<Vol> vols = volRepository.findAll();
    List<VolDTO> collect = vols.stream().map(vol ->
dtoMapper.fromVol(vol)).collect(Collectors.toList());
    return collect;
    return null;
}

@Override
public VolDTO getVol(Long volId) {
    Vol vol = volRepository.findById(volId).orElse(null);
    return dtoMapper.fromVol(vol);
}

@Override
public VolDTO updateVol(VolDTO volDTO) {
    Vol vol = dtoMapper.fromVolDTO(volDTO);
    Vol savedVol = volRepository.save(vol);
    return dtoMapper.fromVol(savedVol);
}

@Override
public void deleteVol(Long volId) {
    volRepository.deleteById(volId);
}

@Override
public ReservationDTO saveReservation(ReservationDTO reservationDTO) {
    Reservation reservation =
dtoMapper.fromReservationDTO(reservationDTO);
    Reservation savedReservation =
reservationRepository.save(reservation);
    return dtoMapper.fromReservation(savedReservation);
}

@Override
public List<ReservationDTO> listReservations() {
    List<ReservationDTO> collect =
reservationRepository.findAll().stream().map(reservation ->
dtoMapper.fromReservation(reservation)).collect(Collectors.toList());
    return collect;
}

@Override
public ReservationDTO getReservation(Long reservationId) {
    Reservation reservation =

```

```

reservationRepository.findById(reservationId).orElse(null);
        return dtoMapper.fromReservation(reservation);
    }

    @Override
    public ReservationDTO updateReservation(ReservationDTO reservationDTO) {
        Reservation reservation =
            dtoMapper.fromReservationDTO(reservationDTO);
        Reservation savedReservation =
            reservationRepository.save(reservation);
        return dtoMapper.fromReservation(savedReservation);
    }

    @Override
    public void deleteReservation(Long reservationId) {
        reservationRepository.deleteById(reservationId);
    }
}

```

- Créer le Web service RESTful qui permet de gérer les Vols

**VolController :**

```

package akasmiau.ouassima.akasmiau_ouassima.web;

import akasmiau.ouassima.akasmiau_ouassima.dao.entities.Vol;
import akasmiau.ouassima.akasmiau_ouassima.dtos.VolDTO;
import akasmiau.ouassima.akasmiau_ouassima.services.ReservationService;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
public class VolController {
    private ReservationService reservationService;

    public VolController(ReservationService reservationService) {
        this.reservationService = reservationService;
    }

    @GetMapping("/vols")
    public List<VolDTO> vols() {
        return reservationService.listVols();
    }

    @GetMapping("/vols/{id}")
    public VolDTO getVol(@PathVariable(name = "id") Long volId) {
        return reservationService.getVol(volId);
    }

    @PostMapping("vols")
    public VolDTO saveVol(@RequestBody VolDTO volDTO) {
        return reservationService.saveVol(volDTO);
    }

    @PutMapping("/vols/{volId}")
    public VolDTO updateVol(@PathVariable Long volId, @RequestBody VolDTO
volDTO) {
        volDTO.setId(volId);
        return reservationService.updateVol(volDTO);
    }
}

```



```

    }
    @DeleteMapping("/vols/{id}")
    public void deleteVol(@PathVariable Long id) {
        reservationService.deleteVol(id);
    }
}

```

## - Créer le Web service RESTful qui permet de gérer les réservations

### ReservationController :

```

package akasmiou.ouassima.akasmiou_ouassima.web;

import akasmiou.ouassima.akasmiou_ouassima.dtos.ReservationDTO;
import akasmiou.ouassima.akasmiou_ouassima.services.ReservationService;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
public class ReservationController {

    private ReservationService reservationService;

    public ReservationController(ReservationService reservationService) {
        this.reservationService = reservationService;
    }

    @GetMapping("/reservations")
    public List<ReservationDTO> reservations() {
        return reservationService.listReservations();
    }

    @GetMapping("/reservations/{id}")
    public ReservationDTO getReservation(@PathVariable(name = "id") Long reservationId) {
        return reservationService.getReservation(reservationId);
    }

    @PostMapping("reservations")
    public ReservationDTO saveReservation(@RequestBody ReservationDTO reservationDTO) {
        return reservationService.saveReservation(reservationDTO);
    }

    @PutMapping("/reservations/{reservationId}")
    public ReservationDTO updateReservation(@PathVariable Long reservationId, @RequestBody ReservationDTO reservationDTO) {
        reservationDTO.setId(reservationId);
        return reservationService.updateReservation(reservationDTO);
    }

    @DeleteMapping("/reservations/{id}")
    public void deleteReservation(@PathVariable Long id) {
        reservationService.deleteReservation(id);
    }
}

```

## - Générer la documentation SWAGGER des API RESTful

The screenshot shows the Swagger UI interface for an OpenAPI definition. The browser address bar indicates the URL is `localhost:8086/swagger-ui/index.html`. The page title is "OpenAPI definition" with a version indicator "v0 OASS". Below the title, there is a "Servers" section with a dropdown menu showing "http://localhost:8086 - Generated server url". The main content area displays two controllers: "vol-controller" and "reservation-controller". Each controller has a list of endpoints with their respective HTTP methods (PUT, GET, POST, DELETE) and path templates. For example, the "vol-controller" has endpoints like `PUT /vols/{volId}`, `GET /vols`, `POST /vols`, `GET /vols/{id}`, and `DELETE /vols/{id}`. The "reservation-controller" has endpoints like `PUT /reservations/{reservationId}`, `GET /reservations`, `POST /reservations`, `GET /reservations/{id}`, and `DELETE /reservations/{id}`.

### Méthode Get vol By Id :

The screenshot shows the Swagger UI interface for the `GET /vols/{id}` endpoint. The "Parameters" section lists a required parameter `id` of type `integer(int64)` with a value of `1`. Below the parameters, there is an "Execute" button and a "Clear" button. The "Responses" section shows the response body for a 200 status code. The response body is a JSON object containing flight details:

```
{
  "id": 1,
  "nom": "vol1",
  "dateDepart": "2023-05-22",
  "dateArrivee": "2023-05-22",
  "avionDTO": {
    "id": 1,
    "nom": "avion1",
    "nbPlace": 100
  },
  "voyageDTO": {
    "id": 1,
    "date": "2023-05-22"
  }
}
```

## Méthode Get pour récupérer l'ensemble des vols :

**GET** /vols

Parameters

No parameters

ExecuteClear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8086/vols' \
  -H 'accept: */*'
```

Request URL

http://localhost:8086/vols

Server response

CodeDetails

200

Response body

```
[
  {
    "id": 1,
    "nom": "vol1",
    "dateDepart": "2023-05-22",
    "dateArrivee": "2023-05-22",
    "avionDTO": {
      "id": 1,
      "nom": "avion1",
      "nbPlace": 100
    },
    "voyageDTO": {
      "id": 1,
      "date": "2023-05-22"
    }
  },
  {
    "id": 2,
    "nom": "vol2",
    "dateDepart": "2023-05-22",
    "dateArrivee": "2023-05-22",
    "avionDTO": {
      "id": 2,
      "nom": "avion2",
      "nbPlace": 300
    }
  }
]
```

## Méthode Delete vol:

**DELETE** /vols/{id}

Parameters

Cancel

Name	Description
id * required	
integer(\$int64)	3
(path)	

ExecuteClear

Responses

Curl

```
curl -X 'DELETE' \
  'http://localhost:8086/vols/3' \
  -H 'accept: */*'
```

Request URL

http://localhost:8086/vols/3

Server response

CodeDetails

200

Response headers

```
connection: keep-alive
content-length: 0
date: Sat, 24 Jun 2023 23:47:54 GMT
keep-alive: timeout=60
```

Responses

Code	Description	Links
200	OK	No links

## Méthode Get pour récupérer l'ensemble des réservations :

The screenshot shows a REST client interface with the following sections:

- Parameters:** A tab with "No parameters" and a "Cancel" button.
- Execute:** A blue button to execute the request.
- Clear:** A button to clear the request.
- Responses:** A section showing the response details.
  - Code:** 200
  - Details:** A tab showing the response body.
  - Response body:** A JSON array of two reservation objects.

```
{
  "id": 1,
  "date": "2023-05-22",
  "siteReservation": "www.reservation.com",
  "clientD10": {
    "id": 1,
    "nom": "akasmiou",
    "prenom": "ouassima",
    "email": "ouassima@gmail.com"
  },
  "voyageD10": {
    "id": 1,
    "date": "2023-05-22"
  }
},
{
  "id": 2,
  "date": "2023-05-22",
  "siteReservation": "www.reservation.com",
  "clientD10": {
    "id": 2,
    "nom": "akasmiou",
    "prenom": "mohamed",
    "email": "mohamed@gmail.com"
  },
  "voyageD10": {
    "id": 2,
    "date": "2023-05-22"
  }
}
}
```

## Méthode Get réservation By Id :

The screenshot shows a REST client interface with the following sections:

- Parameters:** A tab with a "Cancel" button.
  - Name:** id \* required
  - Description:** integer(\$int64) (path)
  - Value:** 1
- Execute:** A blue button to execute the request.
- Clear:** A button to clear the request.
- Responses:** A section showing the response details.
  - Code:** 200
  - Details:** A tab showing the response body.
  - Response body:** A JSON object representing a single reservation.

```
{
  "id": 1,
  "date": "2023-05-22",
  "siteReservation": "www.reservation.com",
  "clientD10": {
    "id": 1,
    "nom": "akasmiou",
    "prenom": "ouassima",
    "email": "ouassima@gmail.com"
  },
  "voyageD10": {
    "id": 1,
    "date": "2023-05-22"
  }
}
```

## Méthode Delete réservation:

**DELETE** /reservations/{id}

**Parameters**

Name	Description
id * required integer(\$int64) (path)	3

**Execute** **Clear**

**Responses**

**Curl**

```
curl -X 'DELETE' \
  'http://localhost:8086/reservations/3' \
  -H 'accept: */*'
```

**Request URL**

```
http://localhost:8086/reservations/3
```

**Server response**

Code	Details
200	<b>Response headers</b> connection: keep-alive content-length: 0 date: Sun, 25 Jun 2023 00:10:19 GMT keep-alive: timeout=60

**Responses**

Code	Description	Links
200	OK	No links

- Tester les Web service avec un client REST comme Postman

## Méthode Get (récupérer l'ensemble des vols) :

OpenAPI definition / vols / vols

**GET** {{baseUrl}}/vols **Send**

Params Authorization Headers (7) Body Pre-request Script Tests Settings **Cookies**

**Body** Cookies Headers (5) Test Results

200 OK 43 ms 495 B Save as Example

**Pretty** Raw Preview Visualize JSON

```
1 {
2   {
3     "id": 1,
4     "nom": "vol1",
5     "dateDepart": "2023-05-22",
6     "dateArrivee": "2023-05-22",
7     "avionDTO": {
8       "id": 1,
9       "nom": "avion1",
10      "nbPlace": 100
11    },
12    "voyageDTO": {
13      "id": 1,
14      "date": "2023-05-22"
15    }
16  },
17  {
18    "id": 2,
19    "nom": "vol2",
```

## Méthode Get vol By Id :

HTTP / vols / {id} / get Vol

GET {{baseUrl}}/vols/:id

Send

Params • Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Path Variables

Key	Value	Description
id	2	Description

Body Cookies Headers (5) Test Results 200 OK 30 ms 328 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "nom": "vol2",
4   "dateDepart": "2023-05-22",
5   "dateArrivee": "2023-05-22",
6   "avionDT0": {
7     "id": 3,
8     "nom": "avion3",
9     "nbPlace": 200
10  },
11  "voyageDT0": {
12    "id": 2,
13    "date": "2023-05-22"
14  }
```

## Méthode Delete vol:

HTTP / vols / {id} / delete Vol

DELETE {{baseUrl}}/vols/:id

Send

Params • Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Path Variables

Key	Value	Description
id	2	Description

Body Cookies Headers (4) Test Results 200 OK 24 ms 23 B Save as Example

Pretty Raw Preview Visualize Text

```
1
```

## Méthode Get pour récupérer l'ensemble des réservations :

The screenshot shows a REST client interface with a GET request to `{{baseUrl}}/reservations`. The response is a JSON array of reservation objects. The response status is 200 OK, with a response time of 462 ms and a body size of 748 B.

```
1 [
2   {
3     "id": 1,
4     "date": "2023-05-22",
5     "siteReservation": "www.reservation.com",
6     "clientDT0": {
7       "id": 1,
8       "nom": "akasmiau",
9       "prenom": "ouassima",
10      "email": "ouassima@gmail.com"
11    },
12    "voyageDT0": {
13      "id": 1,
14      "date": "2023-05-22"
15    }
16  }
17 ]
```

## Méthode Get réservation By Id :

The screenshot shows a REST client interface with a GET request to `{{baseUrl}}/reservations/:id`. The response is a JSON object representing a single reservation. The response status is 200 OK, with a response time of 16 ms and a body size of 358 B.

```
1 {
2   "id": 2,
3   "date": "2023-05-22",
4   "siteReservation": "www.reservation.com",
5   "clientDT0": {
6     "id": 2,
7     "nom": "akasmiau",
8     "prenom": "mohamed",
9     "email": "mohamed@gmail.com"
10  },
11  "voyageDT0": {
12    "id": 2,
13    "date": "2023-05-22"
14  }
15 }
```

## Méthode Delete réservation:

The screenshot shows a REST client interface with a DELETE request configured. The URL is `{{baseUrl}}/reservations/:id`. The response status is 200 OK, with a response time of 2 ms and a body size of 123 B. The response is displayed in the 'Body' tab, which is currently empty.

**Request Configuration:**

- Method: DELETE
- URL: `{{baseUrl}}/reservations/:id`
- Path Variables:

Key	Value	Description
id	3	Description

**Response:** 200 OK (2 ms, 123 B)

**Body:**

```
1
```