**Département Mathématiques et Informatique**

**Compte rendu**

**Filière :**

**« Ingénierie Informatique : Big Data et Cloud Computing »**

**II-BDCC**

**Examen JEE**

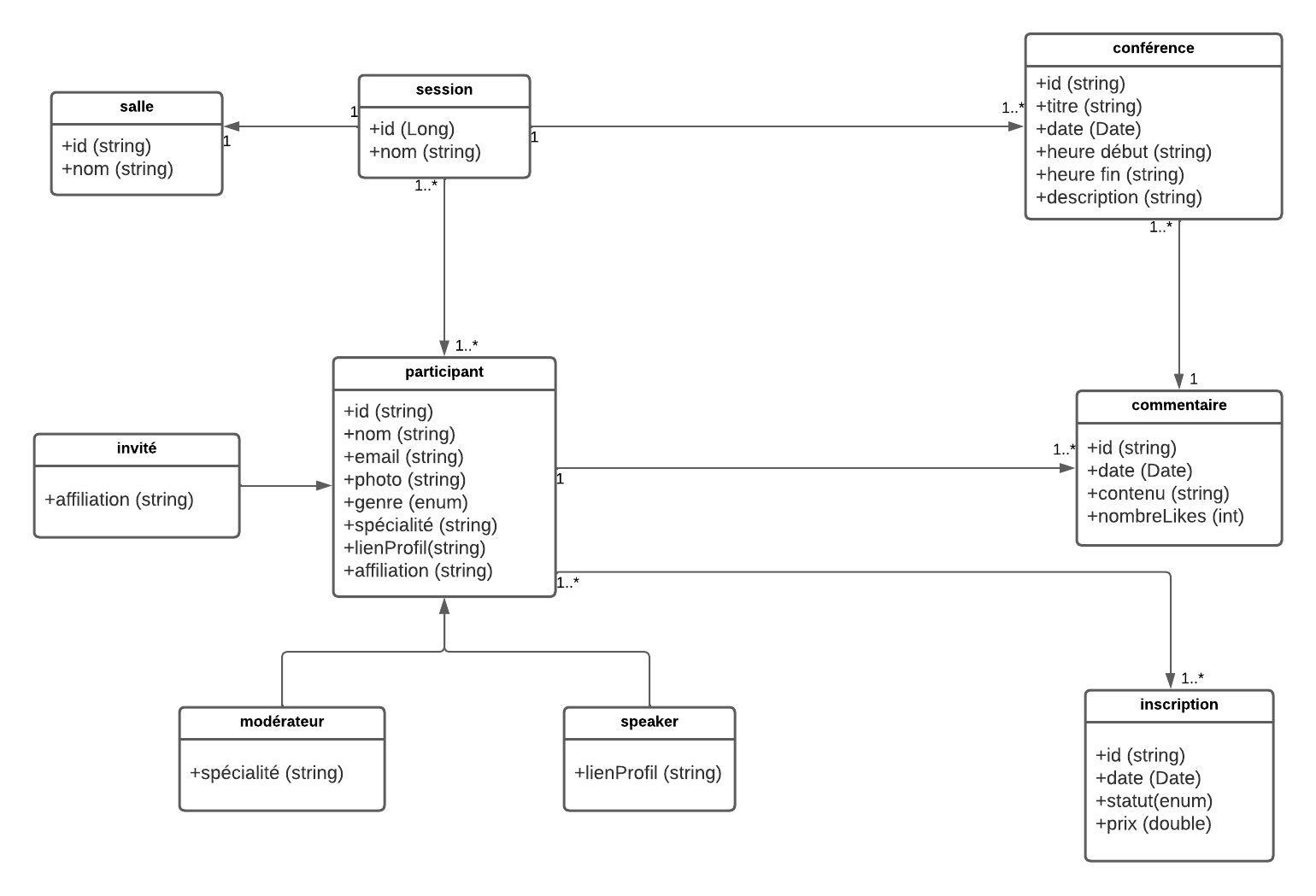
Réalisé par : Salma Khadidi

**Année Universitaire : 2021-2022**

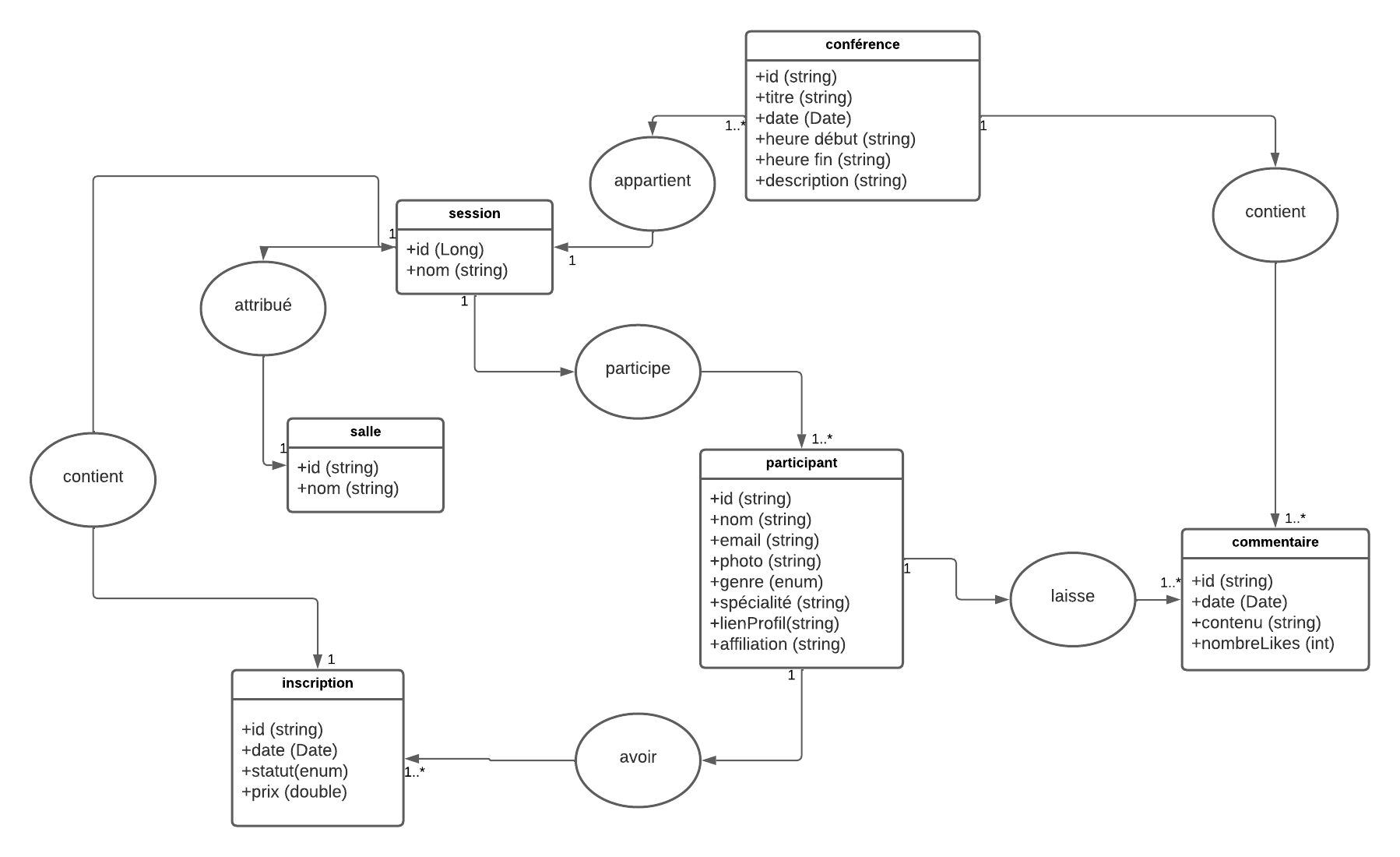
Diagramme de classe

Diagramme de classe

**diagramme de classe**

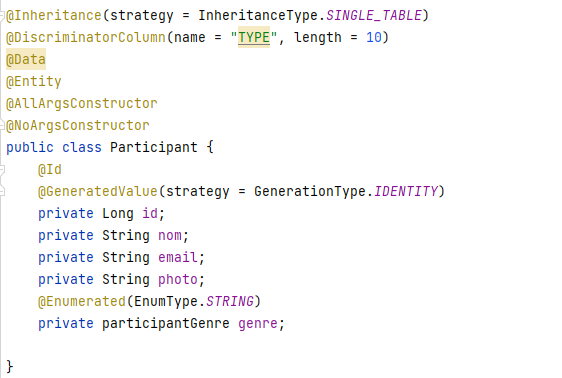
****

**MLD**

****

**Le dossier Entities**

**Entity participant**

****

**Entity invité**

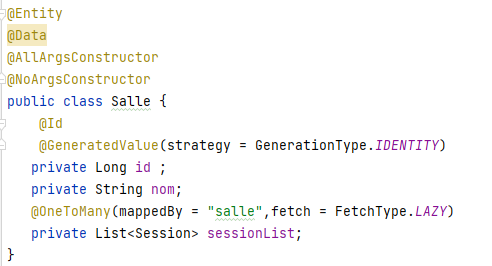
****

**Entity speaker**

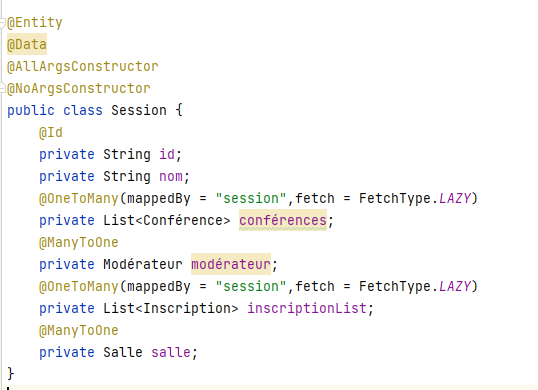


**Entity modérateur**

**Entity salle**



**Entity session**



**Entity conference**



**Entity inscription**

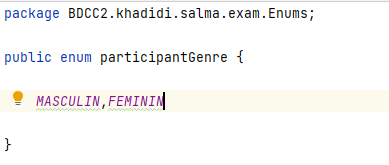


**Entity commentaire**

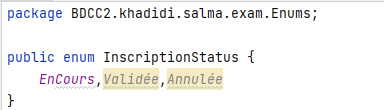


**Le dossier enums**

**Participant enumération**



**Inscription enumération**



**Le dossier Repositories**

package BDCC2.khadidi.salma.exam.Repositories;  
  
import BDCC2.khadidi.salma.exam.Entities.Commentaire;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface CommentaireRepository extends JpaRepository<Commentaire, String> {  
}

**ConferenceRepository**

package BDCC2.khadidi.salma.exam.Repositories;  
  
import BDCC2.khadidi.salma.exam.Entities.Conférence;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface ConférenceRepository extends JpaRepository<Conférence,String> {  
}

**InscriptionRepository**

package BDCC2.khadidi.salma.exam.Repositories;  
  
import BDCC2.khadidi.salma.exam.Entities.Inscription;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface InscriptionRepository extends JpaRepository<Inscription,String> {  
}

**ParticipantRepository**

package BDCC2.khadidi.salma.exam.Repositories;  
  
import BDCC2.khadidi.salma.exam.Entities.Participant;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface ParticipantRepository extends JpaRepository<Participant, Long> {  
}

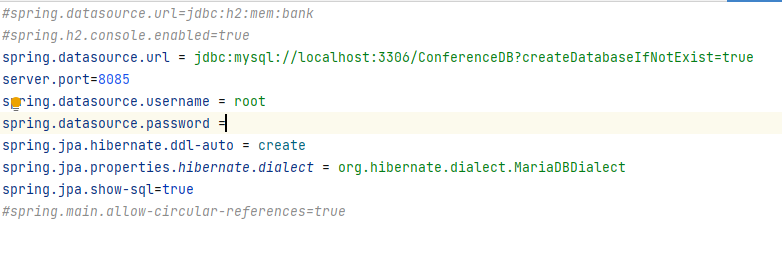
**SalleRepository**

package BDCC2.khadidi.salma.exam.Repositories;  
  
import BDCC2.khadidi.salma.exam.Entities.Salle;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface SalleRepository extends JpaRepository<Salle, String> {  
}

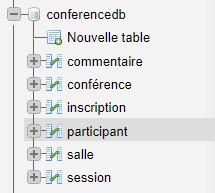
**SessionRepository**

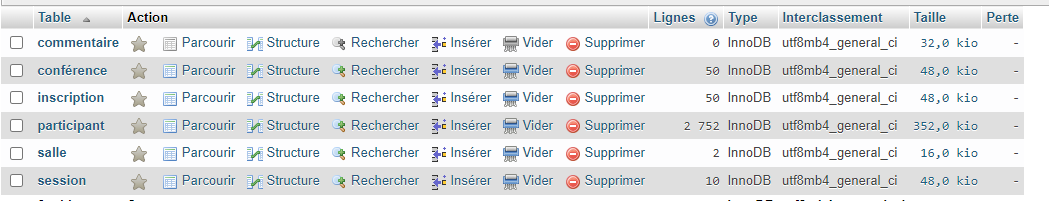
package BDCC2.khadidi.salma.exam.Repositories;  
  
import BDCC2.khadidi.salma.exam.Entities.Session;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface SessionRepository extends JpaRepository<Session, String> {  
}

**Le fichier de connexion à la base de données**

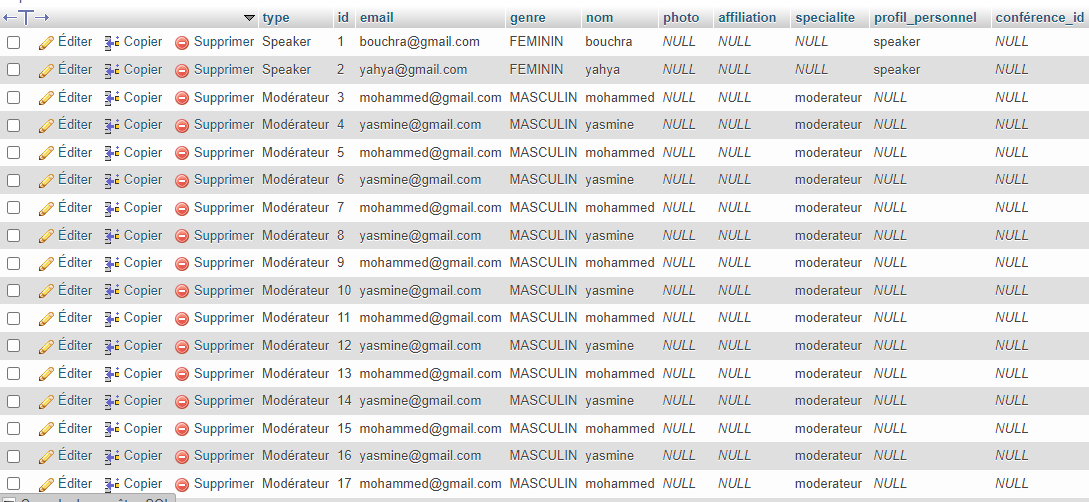


**La base de données**





**La table participant**

****

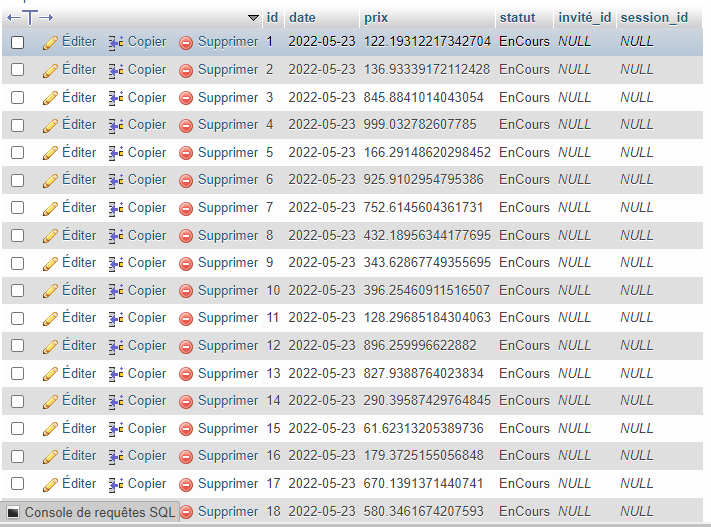
**La table salle**

****

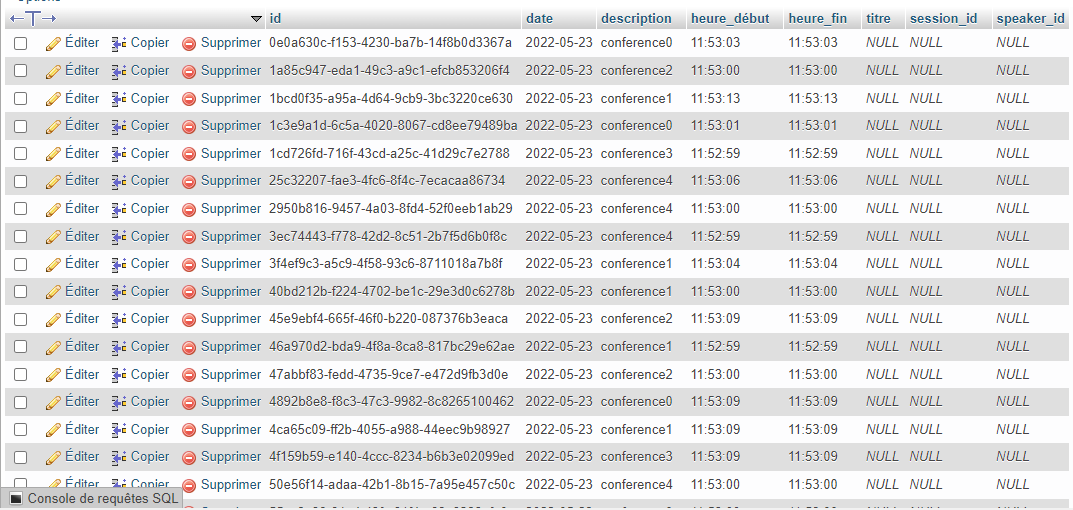
**La table session**

****

**La table inscription**

****

**La table conférence**

****

**La table commentaire**

****

**Le dossier Service**

**L’interface ConferenceService**

package BDCC2.khadidi.salma.exam.Service;  
  
import BDCC2.khadidi.salma.exam.DTOS.\*;  
import BDCC2.khadidi.salma.exam.Entities.Invité;  
import BDCC2.khadidi.salma.exam.Entities.Modérateur;  
import BDCC2.khadidi.salma.exam.Entities.Speaker;  
  
import java.util.List;  
  
public interface ConferenceService {  
 ConferenceDTO getConféranceDTO(String id);  
 List<ConferenceDTO> listConférence();  
 ConferenceDTO updateConférence(String id);  
 ConferenceDTO deleteConférence(String id);  
  
 List<ParticipantDTO> listParticipant();  
 InvitéDTO saveInvité(Invité invité);  
 ModérateurDTO saveModérateur(Modérateur modérateur);  
 SpeakerDTO saveSpeaker(Speaker speaker);  
 ParticipantDTO getParticipant(Long id);  
 SpeakerDTO getSpeakerDTO();  
 ModérateurDTO getModérateur();  
 InvitéDTO getInvité();  
 ParticipantDTO updateParticipant(Long id);  
 ParticipantDTO deleteParticipant(Long id);  
 List<CommentaireDTO> listCommentaires();  
  
 List<SessionDTO> listSessions();  
 SessionDTO getSession(String id);  
}

**ConferenceServiceImplementation**

package BDCC2.khadidi.salma.exam.Service;  
  
import BDCC2.khadidi.salma.exam.DTOS.\*;  
import BDCC2.khadidi.salma.exam.Entities.Invité;  
import BDCC2.khadidi.salma.exam.Entities.Modérateur;  
import BDCC2.khadidi.salma.exam.Entities.Speaker;  
  
import java.util.List;  
  
public class ConferenceServiceImpl implements ConferenceService {  
  
 @Override  
 public ConferenceDTO getConféranceDTO(String id) {  
 return null;  
 }  
  
 @Override  
 public List<ConferenceDTO> listConférence() {  
 return null;  
 }  
  
 @Override  
 public ConferenceDTO updateConférence(String id) {  
 return null;  
 }  
  
 @Override  
 public ConferenceDTO deleteConférence(String id) {  
 return null;  
 }  
  
 @Override  
 public List<ParticipantDTO> listParticipant() {  
 return null;  
 }  
  
 @Override  
 public InvitéDTO saveInvité(Invité invité) {  
 return null;  
 }  
  
 @Override  
 public ModérateurDTO saveModérateur(Modérateur modérateur) {  
 return null;  
 }  
  
 @Override  
 public SpeakerDTO saveSpeaker(Speaker speaker) {  
 return null;  
 }  
  
 @Override  
 public ParticipantDTO getParticipant(Long id) {  
 return null;  
 }  
  
 @Override  
 public SpeakerDTO getSpeakerDTO() {  
 return null;  
 }  
  
 @Override  
 public ModérateurDTO getModérateur() {  
 return null;  
 }  
  
 @Override  
 public InvitéDTO getInvité() {  
 return null;  
 }  
  
 @Override  
 public ParticipantDTO updateParticipant(Long id) {  
 return null;  
 }  
  
 @Override  
 public ParticipantDTO deleteParticipant(Long id) {  
 return null;  
 }  
  
 @Override  
 public List<CommentaireDTO> listCommentaires() {  
 return null;  
 }  
  
 @Override  
 public List<SessionDTO> listSessions() {  
 return null;  
 }  
  
 public SessionDTO getSession(String id) {  
 return null;  
 }  
}

**Le dossier Mappers**

package BDCC2.khadidi.salma.exam.Mappers;  
  
import BDCC2.khadidi.salma.exam.DTOS.\*;  
import BDCC2.khadidi.salma.exam.Entities.\*;  
import org.springframework.beans.BeanUtils;  
  
public class ConferenceMapper {  
 public InvitéDTO fromInvité(Invité invité){  
 InvitéDTO invitéDTO = new InvitéDTO();  
 BeanUtils.*copyProperties*(invité, invitéDTO);  
 invitéDTO.setType(invité.getClass().getSimpleName());  
 return invitéDTO;  
 }  
  
 public Invité fromInvitéDTO(InvitéDTO invitéDTO){  
 Invité invité = new Invité();  
 BeanUtils.*copyProperties*(invitéDTO, invité);  
 return invité;  
 }  
  
 public ModérateurDTO fromModérateur(Modérateur modérateur){  
 ModérateurDTO modérateurDTO = new ModérateurDTO();  
 BeanUtils.*copyProperties*(modérateur, modérateurDTO);  
 modérateurDTO.setType(modérateur.getClass().getSimpleName());  
 return modérateurDTO;  
 }  
  
 public Modérateur fromModérateurDTO(ModérateurDTO modérateurDTO){  
 Modérateur modérateur = new Modérateur();  
 BeanUtils.*copyProperties*(modérateurDTO, modérateur);  
 return modérateur;  
 }  
  
 public SpeakerDTO fromSpeaker(Speaker speaker){  
 SpeakerDTO speakerDTO = new SpeakerDTO();  
 BeanUtils.*copyProperties*(speaker, speakerDTO);  
 speakerDTO.setType(speaker.getClass().getSimpleName());  
 return speakerDTO;  
 }  
  
 public Speaker fromSpeakerDTO(SpeakerDTO speakerDTO){  
 Speaker speaker= new Speaker();  
 BeanUtils.*copyProperties*(speakerDTO, speaker);  
 return speaker;  
 }  
  
 public SessionDTO fromSession(Session session){  
 SessionDTO sessionDTO = new SessionDTO();  
 BeanUtils.*copyProperties*(session, sessionDTO);  
 return sessionDTO;  
 }  
  
 public Session fromSessionDTO(SessionDTO sessionDTO){  
 Session session= new Session();  
 BeanUtils.*copyProperties*(sessionDTO, session);  
 return session;  
 }  
  
 public ConferenceDTO fromConférence(Conférence conférence){  
 ConferenceDTO conférenceDTO = new ConferenceDTO();  
 BeanUtils.*copyProperties*(conférence,conférenceDTO);  
 conférenceDTO.setSpeaker(fromSpeaker(conférence.getSpeaker()));  
 return conférenceDTO;  
 }  
  
 public Conférence fromConférenceDTO(ConferenceDTO conférenceDTO){  
 Conférence conférence = new Conférence();  
 BeanUtils.*copyProperties*(conférenceDTO, conférence);  
 conférence.setSpeaker(fromSpeakerDTO(conférenceDTO.getSpeaker()));  
 return conférence;  
 }  
}

**La classe Main**

package BDCC2.khadidi.salma.exam;  
  
import BDCC2.khadidi.salma.exam.Entities.\*;  
import BDCC2.khadidi.salma.exam.Enums.InscriptionStatus;  
import BDCC2.khadidi.salma.exam.Enums.participantGenre;  
import BDCC2.khadidi.salma.exam.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.Date;  
import java.util.UUID;  
import java.util.stream.Stream;  
  
@SpringBootApplication  
public class ExamenApplication {  
  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(ExamenApplication.class, args);  
 }  
  
  
 @Bean  
 CommandLineRunner commandLineRunner(ParticipantRepository participantRepository,  
 SalleRepository salleRepository,  
 SessionRepository sessionRepository,  
 InscriptionRepository inscriptionRepository,  
 ConférenceRepository conférenceRepository){  
 return args -> {  
  
  
 Stream.*of*("bouchra", "yahya").forEach(name -> {  
 Speaker speaker=new Speaker();  
 speaker.setNom(name);  
 speaker.setEmail(name + "@gmail.com");  
 speaker.setGenre(Math.*random*()>0.5? participantGenre.*MASCULIN*:participantGenre.*FEMININ*);  
 speaker.setProfilPersonnel("speaker");  
 participantRepository.save(speaker);  
 });  
  
 Stream.*of*("salle1", "salle2").forEach(name -> {  
 Salle salle=new Salle();  
 salle.setNom(name);  
 salleRepository.save(salle);  
 });  
  
 salleRepository.findAll().forEach(acc->{  
 for (int i = 0; i < 5; i++) {  
 Session session=new Session();  
 session.setId(UUID.*randomUUID*().toString());  
 session.setNom("session"+i);  
 sessionRepository.save(session);  
 }  
 });  
  
  
 sessionRepository.findAll().forEach(session->{  
 for (int i = 0; i < 5; i++) {  
 Inscription inscription= new Inscription();  
 inscription.setDate(new Date());  
 inscription.setPrix(Math.*random*()\*1000);  
 inscription.setStatut(InscriptionStatus.*EnCours*);  
 inscriptionRepository.save(inscription);  
 }  
  
 for (int i = 0; i < 5; i++) {  
 Conférence conférence=new Conférence();  
 conférence.setId(UUID.*randomUUID*().toString());  
 conférence.setDate(new Date());  
 conférence.setHeureDébut(new Date());  
 conférence.setHeureFin(new Date());  
 conférence.setDescription("conference"+i);  
 conférenceRepository.save(conférence);  
 }  
  
 conférenceRepository.findAll().forEach(acc->{  
 for (int i = 0; i < 5; i++) {  
 Stream.*of*("mohammed", "yasmine").forEach(name -> {  
 Modérateur modérateur=new Modérateur();  
 modérateur.setNom(name);  
 modérateur.setEmail(name + "@gmail.com");  
 modérateur.setGenre(participantGenre.*MASCULIN*);  
 modérateur.setSpecialite("moderateur");  
 participantRepository.save(modérateur);  
 });  
 }  
 });  
 });  
  
  
 };  
 }  
  
  
}