esprit
Se former autrement
HONORIS UNITED UNIVERSITIES

Architecture des SI II (Spring Framework)

Etude de cas gestion “Gestion Foyer”

UP ASI
Bureaux E204 | E304
Année Universitaire : 2023-2024

Etude De Cas : Gestion Foyer (Énoncé)

Objectifs

Développer et déployer une application web d'entreprise en utilisant le Framework Spring.

- Une application de gestion de foyer universitaire visant à simplifier le processus de réservation des chambres pour les étudiants.
- La relation bidirectionnelle **Universite - Foyer** modélise le fait qu'un foyer peut être associé à une seule universite et qu'une université ne peut avoir qu'un seul foyer (foyer est le child).
- La relation bidirectionnelle **Reservation - Etudiant** modélise le fait qu'un étudiant peut avoir plusieurs réservations et qu'une réservation peut être associé à une liste des étudiants (Etudiant est le child).

Etude De Cas : Gestion Foyer

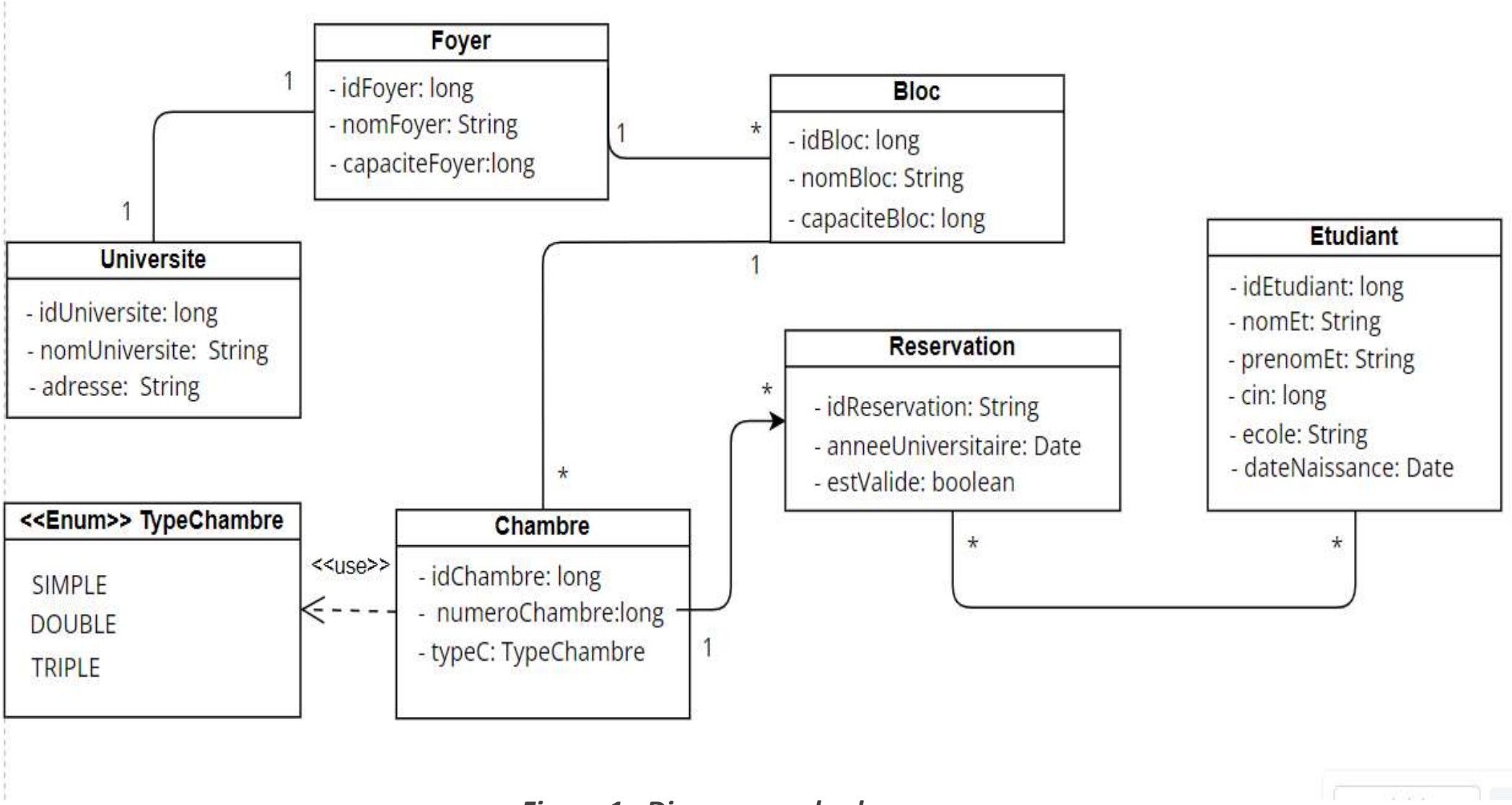

Figure 1 : Diagramme de classes

Travail à faire

Partie 1 Spring Data JPA – Première entité

- Créer les entités se trouvant dans le diagramme des classes (sans les associations) et vérifier qu'ils ont été ajoutés avec succès dans la base de données.

Travail à faire

Partie 2 Spring Data JPA – Le mapping des différentes associations

- Supprimer les tables existantes dans la base de données.
- Créer les associations entre les différentes entités.
- Générer la base de données de nouveau et vérifier que le nombre de tables créées est correct.

Travail à faire

Partie 3 Spring Data JPA CRUD Repository–Le langage JPQL - JPA Repository

Créer les CRUD des différentes **entités indiquées dans les slides suivants en respectant les signatures suivantes**

Entité Foyer

List<Foyer> retrieveAllFoyers();

Foyer addFoyer (Foyer f);

Foyer updateFoyer (Foyer f);

Foyer retrieveFoyer (long idFoyer);

void removeFoyer (long idFoyer);

Travail à faire

Partie 3 Spring Data JPA CRUD Repository–Le langage JPQL - JPA Repository

Créer les CRUD des différentes **entités indiquées dans les slides suivants en respectant les signatures suivantes**

Entité Etudiant

List<Etudiant> retrieveAllEtudiants();

List<Etudiant> addEtudiants (List<Etudiant> etudiants);

Etudiant updateEtudiant (Etudiant e);

Etudiant retrieveEtudiant(long idEtudiant);

void removeEtudiant(long idEtudiant);

Travail à faire

Partie 3 Spring Data JPA CRUD Repository–Le langage JPQL - JPA Repository

Créer les CRUD des différentes **entités indiqués dans les slides suivants en respectant les signatures suivantes**

Entité Bloc

List<Bloc> retrieveBlocs();

Bloc updateBloc (Bloc bloc);

Bloc addBloc (Bloc bloc);

Bloc retrieveBloc (long idBloc);

void removeBloc (long idBloc);

Travail à faire

Partie 3 Spring Data JPA CRUD Repository–Le langage JPQL - JPA Repository

Créer les CRUD des différentes **entités indiqués dans les slides suivants en respectant les signatures suivantes**

Entité Université

List<Universite> retrieveAllUniversities();

Universite addUniversite (Universite u);

Universite updateUniversite (Universite u);

Universite retrieveUniversite (long idUniversite);

Travail à faire

Partie 3 Spring Data JPA CRUD Repository–Le langage JPQL - JPA Repository

Créer les CRUD des différentes **entités indiqués dans les slides suivants en respectant les signatures suivantes**

Entité Chambre

List<Chambre> retrieveAllChambres();

Chambre addChambre(Chambre c);

Chambre updateChambre (Chambre c);

Chambre retrieveChambre (long idChambre);

Travail à faire

Partie 3 Spring Data JPA CRUD Repository–Le langage JPQL - JPA Repository

Créer les CRUD des différentes **entités indiqués dans les slides suivants en respectant les signatures suivantes**

Entité Réservation

List<Reservation> retrieveAllReservation();

Reservation updateReservation (Reservation res);

Reservation retrieveReservation (String idReservation);

Travail à faire

Partie 4 Spring MVC

Exposer les services implémentés dans la partie 3 avec Postman et/ou Swagger pour les tester.

Travail à faire

Partie 5 : Services avancés

On désire affecter un Foyer à une Universite.

Créer un service permettant l'affectation d'un Foyer à une Universite et exposer le en respectant la signature suivante :

public Universite affecterFoyerAUniversite (long idFoyer, String nomUniversite) ;

Travail à faire

Partie 5 : Services avancés

On désire désaffecter un Foyer à une Universite.

Créer un service permettant la désaffectation d'un Foyer à une Universite et exposer le en respectant la signature suivante :

public Universite desaffecterFoyerAUniversite (long idUniversite) ;

Travail à faire

Partie 5 : Services avancés

On désire affecter des Chambres à un Bloc.

Créer un service permettant l'affectation des Chambres à un Bloc et exposer le en respectant la signature suivante :

public Bloc affecterChambresABloc(List<Long> numChambre, long idBloc) ;

Travail à faire

Partie 5 : Services avancés

On désire ajouter à la fois un Foyer ses blocs associés et l'affecter à une université donnée.

- Il faut créer en même temps la liste des blocs (l'entité associée Bloc au Foyer) tout en assurant les affectations nécessaires.

- **N.B :** L'ensemble des objets Blocs liés au Foyer seront inclus et encapsulés avec lui.

Créer le service adéquat et exposer le en respectant la signature suivante :

public Foyer ajouterFoyerEtAffecterAUniversite (Foyer foyer, long idUniversite) ;

Travail à faire

Partie 5 : Services avancés

On désire ajouter une réservation et l'affecter à la fois à une chambre à un étudiant donné.

Au moment de l'ajout de la réservation, vous devez prendre en considérations les consignes suivants :

- **numReservation** doit être sous le format suivant : *numChambre-nomBloc-anneeUniversitaire*
- **estValide**: true

N.B : L'ajout de la réservation se fait que si la capacité maximale de la chambre (selon le type de la chambre SIMPLE, DOUBLE ou TRIPLE) est encore non atteinte.

Créer le service adéquat et exposer le en respectant la signature suivante :

public Reservation ajouterReservation (long idBloc, long cinEtudiant) ;

Travail à faire

Partie 5 : Services avancés

On désire annuler une réservation d'un étudiant donné.

Créer un service permettant d'annuler une réservation selon la cin d'un étudiant donné et exposer le en respectant la signature suivante :

public Reservation annulerReservation (long cinEtudiant) ;

N.B : L'annulation de la réservation permet de :

- Mettre à jour l'état de la réservation (**estValide**: false)
- Désaffecter l'étudiant associé
- Désaffecter la chambre associée et mettre à jour sa capacité

Travail à faire

Partie 5 : Services avancés

Créer un service qui affiche les chambres d'une université spécifique en utilisant **son nom comme attribut unique**, tout en respectant la signature suivante :

public List<Chambre> getChambresParNomUniversite(String nomUniversite) ;

Travail à faire

Partie 5 : Services avancés

On souhaite récupérer les chambres d'un bloc donné selon leur type.

Créer un service permettant de lister les chambres d'un bloc selon un type donné en proposant deux solutions différents (JPQL et Keywords) et exposer le en respectant la signature suivante :

public List<Chambre> getChambresParBlocEtType (long idBloc, TypeChambre typeC) ;

Travail à faire

Partie 5 : Services avancés

Créer un service permettant l'affichage des réservations effectuée lors d'une année universitaire donnée et exposer le en respectant la signature suivante :

public List<Reservation> getReservationParAnneeUniversitaireEtNomUniversite(Date anneeUniversite, String nomUniversite) ;

NB: Le nom d'une université est unique

Travail à faire

Partie 5 : Services avancés

Créer un service permettant l'affichage des chambres non réservées ,par typeChambre , appartenant à un foyer donné par son nom, effectué durant l'année universitaire actuelle et exposer le en respectant la signature suivante :

public List<Chambre> getChambresNonReserveParNomUniversiteEtTypeChambre(String nomUniversite,TypeChambre type) ;

Travail à faire

Partie 6 : Scheduler

Créer un service permettant l'affichage des chambres non réservées pendant cette année pour toutes les universite

Travail à faire

Partie 6 : AOP

Créer un aspect permettant l'affichage de temps d'exécution de la méthode ajouter réservation.

Etude De Cas : Gestion Foyer

Si vous avez des questions, n'hésitez pas à nous contacter :

Département Informatique
UP ASI (Architectures des Systèmes d'Information)
Bureaux E204 | E304
Année Universitaire : 2023-2024