Rendu 4

Etape1

Les attributs et opérations de chaque classe :

* *Personne :*

Attributs: nom (string), prenom (string), email (string), mop (string)

Operations: s’authentifier (nom: string, mop: string), consulterinformations (), modifierinformations ()

* *Elève :*

Attributs: id\_eleve (int), progression(string), listeCours (List<Cours>)

Operations : consulterinformations (), modifierinformations (), sinscrireCours (cours : Cours), suivreCours (cours : Cours)

* *Instructeur :*

Attributs : id\_instructeur (int)

Operations : consulterinformations (), modifierinformations (), consulterCours (List<Cours>), creerCours (cours : Cours), modifierCours (Cours : cours), supprimerCours (Cours : cours)

* *Admin :*

Attributs : id\_manager (int)

Operations : consulterEleves (), creerEleve (eleve : Eleve), modifierEleve (eleve : Eleve), supprimerEleve (eleve : Eleve), consulerinstructeur (List<Instructeur>), creeinstructeur (Instructeur : instructeur), modifierinstructeur (Instructeur : instructeur), supprimerinstructeur (Instructeur : instructeur)

* *Cours :*

Attributs : id (int), dateDebut (date), datefin (date) , nomCours (string), id\_instructeur (int), listeEleves (List<Eleve>), listeEvaluations (List<Evaluation>), horraires (Horraire)

Operations : annulerCours (), modifierCours ()

* *Evaluation :*

Attributs : idEval (int), dateEval (date), description (string), note (int), id\_eleve (int)

Operations: ajouterNote (eleve: Eleve, note: int), consulterNotes (List<int>)

* *Horaire :*

Attributs: id (int), jour (string), heureDebut (string), heurefin (string)

Operations: creerHorraire (jour: string, heureDebut: string, heureFin: string), lireHorraire (id: int), mettreAjourHorraire (id: int, jour: string, heureDebut: string, heureFin: string), supprimerHorraire (id: int)

* *Réservation :*

Attributs : id (int), eleve (Eleve), cours (Cours), dateReservation (date)

Operations : ajouterReservation (Reservation : Reservation), supprimerReservation (Reservation : Reservation), modifierReservation (Reservation : Reservation), annulerReservation ()

* *Contact :*

Attributs: nom (string), prenom (string), email (string), message (string)

Operations : envoyerMessage (Contact : Contact), ConsulterMessage (List<Contact>)

* *ConnexionDemande :*

Attributs: email (string), motDePasse (string)

Operations: authentifier (email: string, motDePasse: string), reinitialiserMotDePasse ()

Traduction des relations:

- Admin->Reservation (Association) : admin gère les réservations

- Admin->Contact (Association) : contact est gérer par admin

- Admin->Personne (Héritage) : admin hérite de la classe personne

- ConnexionDemande->Personne (Association) : la classe Personne possède ConnexionDemande

-Instructeur->Personne (Héritage) : Instructeur hérite de la classe Personne

-Evaluation->Eleve (Association) : Eleve est évalué par Evaluation

-Eleve->Cours (Association) : Eleve suit un Cours

-Evaluation->Cours (Association) : Evaluation appartient à un Cours

-Instructeur->Cours (Association) : Instructeur enseigne Cours

-Horraire->Cours (Association) : Horraire appartient à Cours

-Admin->Cours (Association) : Admin gère Cours

-Admin->Instructeur (Association) : Admin consulte Instructeur

-Admin->Eleve (Association): Admin consulte Eleve

Définition de la visibilité, du type et des valeurs par défaut des attributs :

-Les attributs de la classe Personne sont « protected »

-Les attributs des classes restantes sont « private »

-Les méthodes de toutes les classes sont « public »

-Les types des attributs sont : int, string, date, List<>

-Les valeurs par défaut ne sont pas spécifiées dans le diagramme, mais peuvent être définies selon les besoins du système

Définition d’un invariant pour la classe Eleve :

Un invariant pour la classe Eleve pourrait être que la progression d’un élève ne doit pas dépasser 100

En logique propositionnelle, nous pouvons le formuler comme suit :

🡪 Si P représente la progression de l’élève, alors 0 <= P <= 100

Table de décision des tests :

*Table de décision de tests pour l’opération s’authentifier (nom : string, mop : string) :*

|  |  |  |  |
| --- | --- | --- | --- |
| TC | nom | mop | Résultat attendue |
| TC1 | Ahmed | Ahmed8795 | Connexion réussie |
| TC2 |  | Aziz789 | Connexion échoué (nom d’utilisateur vide) |
| TC3 | Rami |  | Connexion échoué (mop vide) |

*TC1 :* authentification réussie avec un nom d’utilisateur valide et un mot de passe valide

*TC2 :* Echec de l’authentification en raison d’un mot d’utilisateur vide, même avec un mot de passe valide

*TC3 :* Echec de l’authentification en raison d’un mot de passe vide, même avec un nom d’utilisateur valide

*Table de décision de tests pour l’opération ajouter\_reservation () :*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC | Condition1 : Date de réservation valide | Condition2 : Place disponible | Condition : Client valide | Résultat attendu |
| TC1 | Oui | Oui | Oui | Réservation ajoutée avec succès |
| TC2 | Oui | Oui | Non | Echec de l’ajout de réservation (client invalide) |
| TC3 | Oui | Non | Oui | Echec de l’ajout de réservation (place non disponible) |
| TC4 | Non | Oui | Oui | Echec de l’ajout de réservation (date de réservation invalide) |
| TC5 | Non | Non | Oui | Echec de l’ajout de réservation (date de réservation invalide et place non disponible |
| TC6 | Oui | Non | Non | Echec de l’ajout de réservation (place non disponible et client invalide) |
| TC7 | Non | Oui | Non | Echec de l’ajout de réservation (date de réservation invalide et client invalide) |
| TC8 | Non | Non | Non | Echec de l’ajout de réservation |

TC1 : La date de réservation est valide, une place est disponible, et le client est valide. Dans ce cas, la réservation devrait être ajoutée avec succès.

TC2 : La date de réservation est valide et une place est disponible, mais le client n'est pas valide. L'ajout de réservation devrait échouer en raison d'un client invalide.

TC3 : La date de réservation est valide, mais aucune place n'est disponible. L'ajout de réservation devrait échouer en raison de l'indisponibilité de places.

TC4 : La date de réservation n'est pas valide, mais une place est disponible et le client est valide. L'ajout de réservation devrait échouer en raison d'une date de réservation invalide.

TC5 : La date de réservation n'est pas valide et aucune place n'est disponible, mais le client est valide. L'ajout de réservation devrait échouer en raison d'une date de réservation invalide et de l'indisponibilité de places.

TC6 : La date de réservation est valide, mais aucune place n'est disponible et le client n'est pas valide. L'ajout de réservation devrait échouer en raison de l'indisponibilité de places et d'un client invalide.

TC7 : La date de réservation n'est pas valide, une place est disponible, mais le client n'est pas valide. L'ajout de réservation devrait échouer en raison d'une date de réservation invalide et d'un client invalide.

TC8 : La date de réservation n'est pas valide, aucune place n'est disponible, et le client n'est pas valide. L'ajout de réservation devrait échouer en raison d'une date de réservation invalide, de l'indisponibilité de places et d'un client invalide.

*Table de décision de tests pour l’opération supprimer Eleve () :*

|  |  |  |  |
| --- | --- | --- | --- |
| *TC* | Condition1 : Eleve existant | Condition2 : Eleve sélectionné pour la suppression | Résultat attendue |
| TC1 | OUI | OUI | Eleve supprimé avec succès |
| TC2 | Non | N/A | Echec de suppression de l’élève (élève inexistant) |
| TC3 | Oui | Non | Echec de suppression de l’élève (élève non sélectionné pour la suppression) |
| TC4 | Non | N/A | Echec de suppression de l’élève (élève inexistant |

TC1 : L'élève existe dans le système et a été sélectionné pour suppression. Dans ce cas, l'élève devrait être supprimé avec succès.

TC2 : L'élève n'existe pas dans le système. Par conséquent, la suppression de l'élève échoue car il n'y a aucun élève à supprimer.

TC3 : L'élève existe dans le système, mais il n'a pas été sélectionné pour suppression. Par conséquent, la suppression de l'élève échoue car l'administrateur n'a pas spécifiquement sélectionné cet élève pour suppression.

TC4 : Comme dans le cas du TC2, l'élève n'existe pas dans le système. La suppression de l'élève échoue pour la même raison que dans le TC2, à savoir qu'il n'y a aucun élève à supprimer.