EXAMEN SESSION PRINCIPALE

Semestre : 1

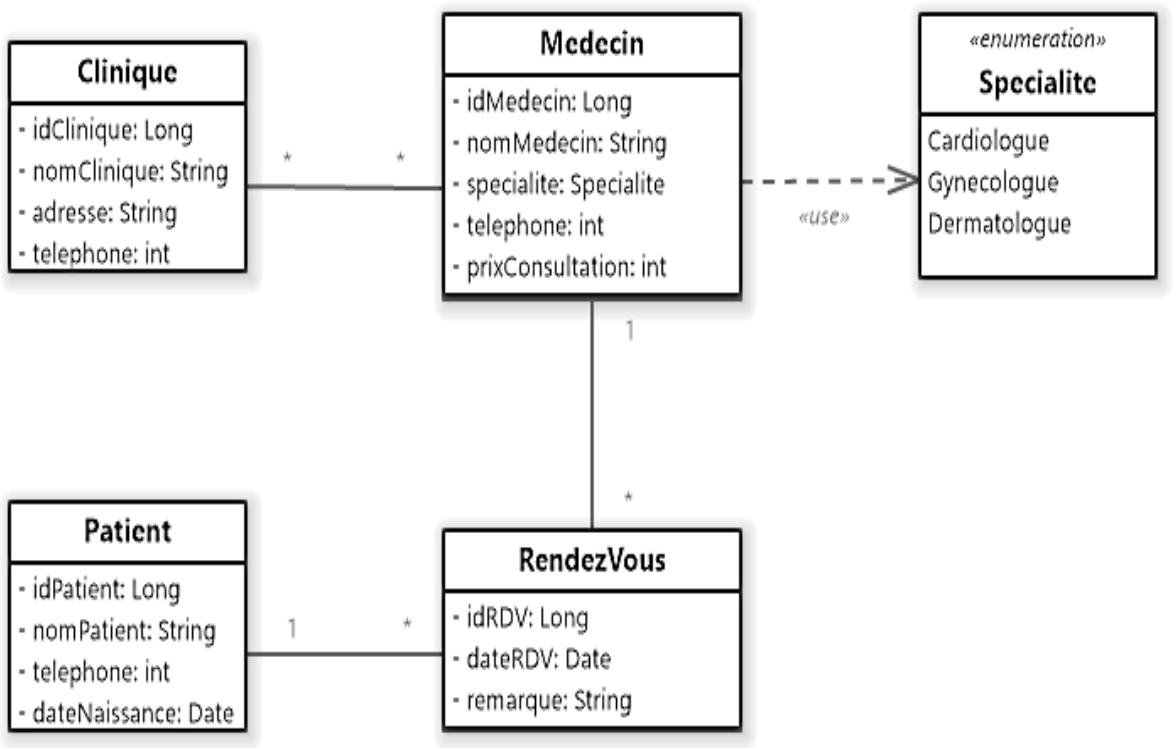
Module : Architecture des Systèmes d'information II		Enseignants : Spring Team
Classe(s) : 4 TWIN/ERP-BI	**Documents autorisés:** OUI	**Internet autorisée:** NON
Date : 10/01/2022	**Durée :** 90 minutes	**Nombre de pages :** 3

La validation de l'épreuve est appliquée sur la base d'un code source exécutable.
Aucun code source non fonctionnel n'est comptabilisé lors de la validation.

On vous propose d'implémenter une application simplifiée de gestion des prises de rendez-vous dans des cliniques.

- Une clinique peut avoir plusieurs Médecins. Un Médecin peut travailler dans plusieurs cliniques.
- Un médecin peut avoir, au sein d'une clinique, l'une des spécialités suivantes : Cardiologue, Gynecologue, Dermatologue.
- Un patient/un médecin peut avoir plusieurs rendez-vous. Mais un rendez-vous spécifique ne peut pas être affecté à un seul médecin et un seul patient.

Ci-dessous le diagramme de classes :



I. (5 pts)

Implémenter les entités qui permettent de générer le schéma de la base de données comme illustré dans le diagramme de classes sachant que :

- Les identifiants des entités sont auto-générés avec la stratégie « IDENTITY ».
- L'association bidirectionnelle Clinique-Medecin, le medecin est le child.
- Les énumérations doivent être stockées en tant que chaines de caractères dans la base de données.

II.

Développer le code nécessaire dans une classe annotée par @RestController qui fait appel aux différents services. (Exposition des services avec Spring REST MVC).

Toutes les méthodes seront testées à travers **Postman** ou **Swagger**.

1. Ajouter les cliniques ci-dessous, en respectant la signature suivante **(1pt)** :

public Clinique addClinique (Clinique clinique)

nomClinique	**adresse**	**telephone**
Taoufik	Manar II	36000000
Elissa	Lac I	49000000

2. Ajouter les médecins ci-dessous en les affectant à une clinique associée, en respectant la signature suivante **(2pts)** :

public Medecin addMedecinAndAssignToClinique (Medecin medecin, Long cliniqueId)

Specialite	**nomMedecin**	**telephone**	**prixConsultation**	**clinique**
Cardiologue	Mohamed Tounsi	70000000	50	Taoufik
Dermatologue	Halima Ben Khalifa	71000000	65	Taoufik
Gynecologue	Iyed Hannachi	72000000	70	Elissa

3. Ajouter les patients ci-dessous, en respectant la signature suivante **(1pt):**
   public Patient addPatient(Patient patient)

nomPatient	**telephone**	**dateNaissance**
Amal Hfaiedh	98333333	1997-02-14
Eya Slimene	96325847	1996-01-02
Borhen khlifa	87459632	1974-08-28

4. Ajouter un nouveau rendez-vous et l'affecter à la fois à un médecin et un patient, en respectant la signature suivante **(2 pts)**:
   public void addRDVAndAssignMedAndPatient(RendezVous rdv, Long idMedecin, Long idPatient)

dateRDV	remarque	medecin	patient
2022-02-02	Patient malade	Mohamed Tounsi	Amal Hfaiedh
2022-02-15	Patient sain	Mohamed Tounsi	Amal Hfaiedh
2022-01-08	Patient sain	Iyed Hannachi	Borhen khlifa
2022-03-02	Patient sain	Mohamed Tounsi	Eya Slimene

5. Afficher la liste des rendezVous de la clinique Taoufik dont la spécialité est « Cardiologue », en respectant la signature suivante **(2pts)**:
   public List<RendezVous> getRendezVousByCliniqueAndSpecialite(Long idClinique,Specialite specialite)

6. Afficher le nombre des rendezVous d'un médecin, en respectant la signature suivante **(2pts)** :
   public int getNbrRendezVousMedecin(Long idMedecin)

7. Nous souhaitons créer un service programmé automatiquement (scheduled) permettant d'afficher tous les rendez-vous dont la date de rendez-vous est supérieure à la date système, comme le montre la figure 1.
   Créer le service qui permet d'afficher les rendez-vous concernés toutes les 30 secondes en respectant la signature suivante **(2pts)** :
   public void retrieveRendezVous()

```
La liste des RendezVous : 2022-02-02 : Medecin :Mohamed Tounsi : Patient :Amal Hfaiedh
La liste des RendezVous : 2022-02-15 : Medecin :Mohamed Tounsi : Patient :Amal Hfaiedh
La liste des RendezVous : 2022-03-02 : Medecin :Mohamed Tounsi : Patient :Eya Slimene
```

Figure 1 Message à afficher

8. Nous souhaitons calculer le revenu d'un médecin selon le nombre de ses rendez-vous planifiés entre deux dates. Créer le service qui permet de faire ce calcul en respectant la signature suivante au niveau de RestController **(2pts) :**

```
public int getRevenuMedecin (
        @PathVariable("idMedecin") Long idMedecin,
        @PathVariable("startDate") @DateTimeFormat(pattern = "yyyy-MM-dd") Date startDate,
        @PathVariable("endDate") @DateTimeFormat(pattern = "yyyy-MM-dd")Date endDate
                                )
```

Exemple à tester:

startDate = 2022-01-01 , endDate = 2022-12-30
Le revenu du médecin Mohamed Tounsi est = 150dt.

9. Créer un Aspect qui permet d'afficher le message de log suivant « méthode exécutée » dans la console après l'exécution des méthodes de la couche service qui commencent par **add.. (1pt)**

Bon Courage