Projet en vue de l'évaluation finale

C# avancé

1. Equipe

Le projet est réalisé par groupe de 2 étudiants. Il s'agit d'un travail de groupe ou chacun doit fournir la moitié de l'effort. Les équipes doivent être inscrites dans le fichier Excel sur Teams réservé à cet effet (Equipes.xlsx).

2. Examen oral

Le projet est le support principal de l'examen. Le projet sera défendu par les deux étudiants simultanément. Durant cette défense, il pourra être demandé aux étudiants de modifier individuellement le code afin de vérifier les compétences de chacun. En cas d'échec en janvier, le projet devra être finalisé pour le mois de septembre, l'examen écrit devant également être représenté.

3. Dépôt

Le code doit être déposé sur Teams par les deux membres de l'équipe dans l'espace devoir de Teams. Le code doit contenir tout ce qu'il faut pour le faire fonctionner en format .zip.

4. Les délais

La veille du jours de l'examen, la date sera définie dès que la date de l'examen sera connue. N'attendez pas trop pour vous y mettre, ça ne devrait pas vous demander plus d'une journée de travail chacun, l'espace labo est suffisant pour terminer le code.

5. Objectifs pédagogiques

À l'issue de ce projet, vous serez capable de :

- Concevoir et implémenter une architecture REST cohérente
- Modéliser une base de données relationnelle avec Entity Framework Core
- Gérer les relations entre entités (One-to-Many, Many-to-Many)
- Implémenter les opérations CRUD complètes
- Appliquer les bonnes pratiques de développement d'API (Mapper & DTO)
- Gérer les migrations de base de données
- Valider les données et gérer les erreurs

6. Dépôt Git du squelette du projet

Un projet est déposé sur GIT pour vous aider à démarrer. Il est structuré à l'aide du pattern Clean Architecture. Un fichier readme.md permet de comprendre la structure et ou créer vos classes, il est indispensable de le lire attentivement.

Dépôt git : https://github.com/kwilvers/KindomHospital.git

7. Modélisation clinique (C# / EF Core / ASP.NET Core)

Cette modélisation décrit les entités principales du système clinique, adaptées à un projet ASP.NET Core utilisant Entity Framework Core. Chaque entité est définie avec ses types .NET, contraintes, longueurs et relations.

Spécialité (SPECIALTY)

Rôle : référentiel des domaines médicaux (Cardiologie, Dermatologie, etc.).
Id : int (clé primaire).
Name : string (MaxLength 30, Required, Unique).
Relation : 1 Spécialité → N Médecins.

Médecin (DOCTOR)

Rôle : professionnel de santé rattaché à une spécialité.
Id : int (clé primaire).
SpecialtyId : int (FK Required → Specialty.Id).
LastName / FirstName : string (MaxLength 30, Required).
Relations : N Médecins → 1 Spécialité ; 1 Médecin → N Consultations et N Ordonnances.

Patient (PATIENT)

Rôle : personne suivie par le cabinet/labo.
Id : int (clé primaire).
LastName / FirstName : string (MaxLength 30, Required).
BirthDate : DateOnly ou DateTime (Required).
Relations : 1 Patient → N Consultations et N Ordonnances.

Consultation (CONSULTATION)

Rôle : rencontre datée entre un médecin et un patient, avec un motif.
Id : int (clé primaire).
DoctorId / PatientId : int (FK Required).
Date : DateOnly ou DateTime (Required).
Hour : TimeOnly ou TimeSpan (Required).
Reason : string (MaxLength 100, Optional).
Relations : 1 Consultation → 0..N Ordonnances.

Médicament (MEDICAMENT)

Rôle : catalogue des produits prescrits.
Id : int (clé primaire).

Name : string (MaxLength 100, Required, Unique).
DosageForm : string (MaxLength 30, Required).
Strength : string (MaxLength 30, Required).
AtcCode : string (MaxLength 20, Optional).
Relation : 1 Médicament → N Lignes d'ordonnance.

Ordonnance (ORDONNANCE)

Rôle : prescription émise par un médecin pour un patient.
Id : int (clé primaire).
DoctorId / PatientId : int (FK Required).
ConsultationId : int? (FK Optionnelle).
Date : DateOnly ou DateTime (Required).
Notes : string (MaxLength 255, Optional).
Relation : 1 Ordonnance → N Lignes d'ordonnance.

Ligne d'ordonnance (ORDONNANCE_LIGNE)

Rôle : Relation qui décrit le détail d'un médicament prescrit sur une ordonnance.
Id : int (clé primaire).
OrdonnanceId / MedicamentId : int (FK Required).
Dosage / Frequency / Duration : string (50/50/30, Required).
Quantity : int > 0 (Required).
Instructions : string (MaxLength 255, Optional).
Relation : 1 Ordonnance ↔ N Lignes ; 1 Médicament ↔ N Lignes.

8. Données minimales (de démo)

Spécialités

- Voir l'annexe 1 au format CSV. Lire le fichier CSV et écrire les données dans la table dans une méthode Seed.

Médicaments variés

- Voir annexe 2 au format CSV. Lire le fichier CSV et écrire les données dans la table dans une méthode Seed.

Médecins

- Créer un minimum de 6 médecins répartis dans différentes spécialités.

Patients

- Créer un minimum de 5 patients dont les dates de naissances sont plausibles.

Consultations

- Créer un minimum de 10 consultations. Les consultations sont réparties entre les patients (0 à 3 consultations par patient). Un patient ne doit pas avoir de consultations.
- Il doit exister une journée avec 2 consultations du même médecin à des heures différentes.

Ordonnances

- Créer au moins 5 ordonnances dont 1 comporte 3 médicaments dont un patient a au moins deux ordonnances.

9. Endpoints CRUD par entité

Ci-dessous vous trouverez la liste des endpoints que vous devez implémenter. Un minimum de 16 des endpoints doivent être implémenté. Les 8 derniers sont en bonus. Organisez les endpoints par contrôleurs.

SPECIALTIES

`GET /api/specialties` – Liste toutes les spécialités
`GET /api/specialties/{id}` – Détail d'une spécialité

DOCTORS

`GET /api/doctors` – Liste tous les médecins
`GET /api/doctors/{id}` – Détail d'un médecin
`POST /api/doctors` – Crée un médecin
`PUT /api/doctors/{id}` – Met à jour un médecin

PATIENTS

`GET /api/patients` – Liste tous les patients
`GET /api/patients/{id}` – Détail d'un patient
`POST /api/patients` – Crée un patient
`PUT /api/patients/{id}` – Met à jour un patient
`DELETE /api/patients/{id}` – Supprime un patient

CONSULTATIONS

`GET /api/consultations` – Liste toutes les consultations
`GET /api/consultations/{id}` – Détail d'une consultation
`POST /api/consultations` – Crée une consultation
`PUT /api/consultations/{id}` – Met à jour une consultation
`DELETE /api/consultations/{id}` – Supprime une consultation

MEDICAMENTS

`GET /api/medicaments` – Liste tous les médicaments
`GET /api/medicaments/{id}` – Détail d'un médicament
`POST /api/medicaments` – Crée un médicament

ORDONNANCES

`GET /api/ordonnances` – Liste toutes les ordonnances
`GET /api/ordonnances/{id}` – Détail d'une ordonnance
`POST /api/ordonnances` – Crée une ordonnance
`PUT /api/ordonnances/{id}` – Met à jour une ordonnance

`DELETE /api/ordonnances/{id}` – Supprime une ordonnance

10. Endpoints relationnels (Obligatoire)

Ci-dessous vous trouverez la liste des endpoints obligatoire que vous devez implémenter. Organisez les endpoints par contrôleurs, veillez à placer ces endpoints dans les contrôleurs existants.

SPECIALTIES ↔ DOCTORS

`GET /api/specialties/{id}/doctors` – Liste les médecins d'une spécialité
`GET /api/doctors/{id}/specialty` – Renvoie la spécialité d'un médecin
`PUT /api/doctors/{id}/specialty/{specialtyId}` – Change la spécialité d'un médecin

DOCTORS ↔ CONSULTATIONS / PATIENTS

`GET /api/doctors/{id}/consultations?from=&to=&` – Liste les consultations d'un médecin
`GET /api/doctors/{id}/patients` – Liste les patients déjà consultés
`GET /api/doctors/{id}/ordonnances?from=&to=` – Liste les ordonnances émises par le médecin entre deux dates

PATIENTS ↔ CONSULTATIONS / DOCTORS / ORDONNANCES

`GET /api/patients/{id}/consultations` – Liste les consultations d'un patient
`GET /api/patients/{id}/ordonnances` – Liste les ordonnances du patient

CONSULTATIONS ↔ ORDONNANCES

`GET /api/consultations/{id}/ordonnances` – Liste les ordonnances liées à une consultation
`POST /api/consultations/{id}/ordonnances` – Crée une ordonnance rattachée à cette consultation
`PUT /api/ordonnances/{id}/consultation/{consultationId}` – Rattache une ordonnance à une consultation
`DELETE /api/ordonnances/{id}/consultation` – Détache l'ordonnance de sa consultation

MÉDICAMENTS (recherche / relationnel)

`GET /api/medicaments/{id}/ordonnances` – Liste des ordonnances où le médicament apparaît

Lignes d'ordonnance

`GET /api/ordonnances/{id}/lignes` – Liste toutes les lignes d'une ordonnance
`POST /api/ordonnances/{id}/lignes` – Ajoute une ou plusieurs lignes
`GET /api/ordonnances/{id}/lignes/{ligneId}` – Détail d'une ligne d'ordonnance
`PUT /api/ordonnances/{id}/lignes/{ligneId}` – Met à jour une ligne
`DELETE /api/ordonnances/{id}/lignes/{ligneId}` – Supprime une ligne

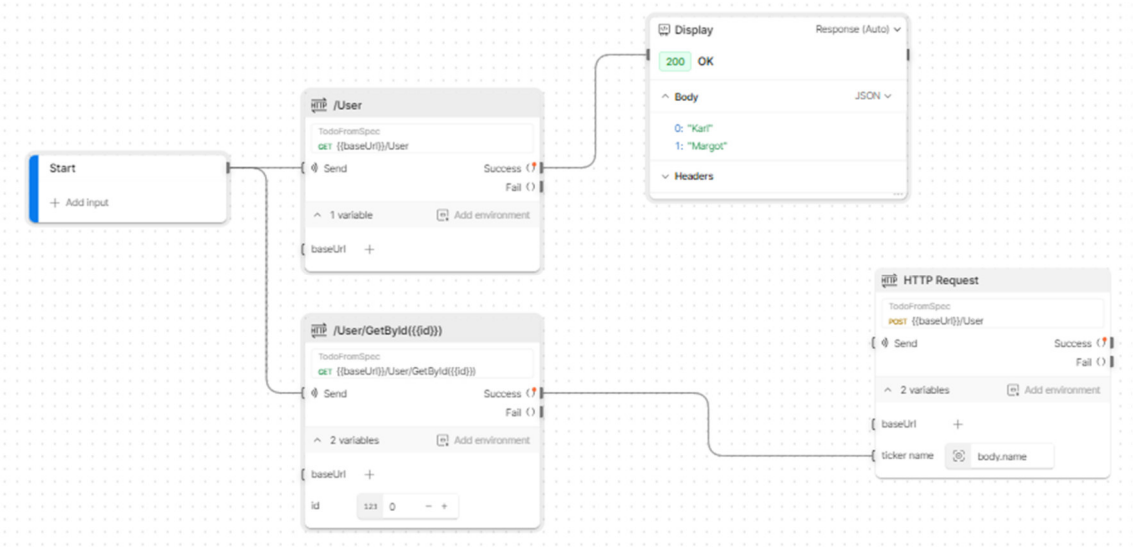
Endpoints utilitaires

`GET /api/consultations?doctorId=&patientId=&from=&to=` – Liste filtrée de consultations par rapport au docteur et/ou patient entre deux dates. Les Ids peuvent être null mais au moins un ne doit pas l'être.

`GET /api/ordonnances?doctorId=&patientId=&from=&to=` – Liste filtrée d'ordonnances par rapport au docteur et/ou patient entre deux dates. Les Ids peuvent être null mais au moins un ne doit pas l'être.

11. Jeu de test

Pour le jour de l'examen, il vous sera demandé d'exécuter un jeu de tests en guise de démonstration. Le jeu de tests est réalisé dans un fichier .http ou dans Postman. En fonction des endpoints réalisés, vous imaginez des séquences de test comprenant des endpoints POST, PUT et DELETE dont la moitié concerne les endpoints relationnels obligatoires. Les modifications doivent être visualisable avant la modification et après la modification.



12. Faux amis et évaluation :

Attention aux faux amis, ChatGpt et compagnie, nous ne pouvons vous interdire de l'utiliser mais vous devez pouvoir expliquer la moindre virgule qui se trouve dans votre code.

Il est conseillé d'utiliser une IA générative pour générer les données par défaut

Durant l'évaluation, vous devez pouvoir identifier une portion de code et en expliquer le contenu. Trouver la fonction ou la partie de code qui effectue une tâche.

Le plagiat sera sanctionné d'un 0, vous forçant à représenter l'ensemble de l'UE en seconde session. Je suis équipé d'un outil me permettant de détecter toutes ressemblances de code sur base d'une analyse sémantique. N'essayé pas je vous prendrai la main dans le sac…

Les commentaires dans le code lors de l'évaluation ne sont pas autorisés à l'exception de la documentation des entêtes de fonctions.

Bon travail.

Annexes :

Annexe 1 : Liste des Spécialités Médicales

Id,Nom,Categorie,Description
1,Chirurgie Générale,Chirurgicale,Interventions abdominales et digestives
2,Chirurgie Orthopédique,Chirurgicale,Os, articulations et ligaments
3,Chirurgie Cardiaque,Chirurgicale,Cœur et vaisseaux
4,Neurochirurgie,Chirurgicale,Cerveau, moelle épinière et nerfs
5,Chirurgie Plastique,Chirurgicale,Reconstructrice et esthétique
6,Chirurgie Thoracique,Chirurgicale,Poumons et cage thoracique
7,Chirurgie Vasculaire,Chirurgicale,Artères et veines
8,Chirurgie Pédiatrique,Chirurgicale,Interventions chez l'enfant
9,Médecine Générale,Médicale,Soins primaires
10,Cardiologie,Médicale,Maladies cardiovasculaires
11,Pneumologie,Médicale,Maladies respiratoires
12,Gastro-entérologie,Médicale,Système digestif
13,Néphrologie,Médicale,Reins et voies urinaires
14,Endocrinologie,Médicale,Hormones et métabolisme
15,Rhumatologie,Médicale,Articulations, os et muscles
16,Neurologie,Médicale,Système nerveux
17,Hématologie,Médicale,Sang et organes hématopoïétiques
18,Oncologie,Médicale,Cancers et tumeurs
19,Gynécologie,Femme et Enfant,Appareil génital féminin
20,Obstétrique,Femme et Enfant,Grossesse et accouchement
21,Pédiatrie,Femme et Enfant,Médecine de l'enfant
22,Néonatologie,Femme et Enfant,Nouveau-nés et prématurés
23,Ophtalmologie,Organes des Sens,Yeux et vision
24,ORL,Organes des Sens,Oreilles, nez et gorge
25,Dermatologie,Organes des Sens,Peau, cheveux et ongles
26,Anesthésiologie,Transversale,Anesthésie et réanimation
27,Radiologie,Transversale,Imagerie médicale
28,Médecine Interne,Transversale,Maladies complexes multi-organes
29,Psychiatrie,Transversale,Santé mentale
30,Gériatrie,Transversale,Médecine des personnes âgées
31,Médecine d'Urgence,Transversale,Urgences et SAMU
32,Médecine du Travail,Transversale,Santé au travail
33,Médecine Légale,Transversale,Expertise judiciaire
34,Allergologie,Médicale,Allergies et immunologie
35,Infectiologie,Médicale,Maladies infectieuses
36,Médecine Physique et Réadaptation,Transversale,Rééducation fonctionnelle
37,Médecine Nucléaire,Transversale,Diagnostic et thérapie par isotopes
38,Anatomo-pathologie,Transversale,Analyse tissulaire et diagnostic
39,Biologie Médicale,Transversale,Analyses de laboratoire
40,Addictologie,Transversale,Dépendances et toxicomanies

Annexe 2 : Liste des médicaments

Id,Name,DosageForm,Strength,AtcCode
1,Paracetamol,Comprimé,500mg,N02BE01
2,Ibuprofene,Comprimé,400mg,M01AE01
3,Amoxicilline,Gélule,500mg,J01CA04
4,Aspirine,Comprimé,100mg,N02BA01
5,Omeprazole,Gélule,20mg,A02BC01
6,Metformine,Comprimé,850mg,A10BA02
7,Atorvastatine,Comprimé,20mg,C10AA05
8,Losartan,Comprimé,50mg,C09CA01
9,Amlodipine,Comprimé,5mg,C08CA01
10,Levothyroxine,Comprimé,75mcg,H03AA01
11,Clopidogrel,Comprimé,75mg,B01AC04
12,Metoprolol,Comprimé,50mg,C07AB02
13,Simvastatine,Comprimé,40mg,C10AA01
14,Tramadol,Gélule,50mg,N02AX02
15,Salbutamol,Inhalateur,100mcg,R03AC02
16,Prednisolone,Comprimé,20mg,H02AB06
17,Ciprofloxacine,Comprimé,500mg,J01MA02
18,Azithromycine,Comprimé,250mg,J01FA10
19,Furosemide,Comprimé,40mg,C03CA01
20,Warfarine,Comprimé,5mg,B01AA03
21,Diazepam,Comprimé,10mg,N05BA01
22,Fluoxetine,Gélule,20mg,N06AB03
23,Sertraline,Comprimé,50mg,N06AB06
24,Insuline Glargine,Injectable,100UI/mL,A10AE04
25,Ramipril,Comprimé,5mg,C09AA05
26,Spironolactone,Comprimé,25mg,C03DA01
27,Digoxine,Comprimé,0.25mg,C01AA05
28,Codeïne,Comprimé,30mg,R05DA04
29,Morphine,Injectable,10mg/mL,N02AA01
30,Héparine,Injectable,5000UI/mL,B01AB01

Annexe 3 : Checklist d'acceptation – Modèle clinique/labo

A. Règles générales (toutes entités)

- ☐ Chaque entité possède une **clé primaire** Id de type entier.
- ☐ Les **chaînes** ont une **longueur max** conforme aux spécifs (voir par entité).
- ☐ Les champs **Required** ne peuvent pas être vides ni blancs (trim + validation).
- ☐ Les **relations** (FK) existent, sont cohérentes et empêchent les orphelins (DeleteBehavior adapté).
- ☐ Les **index** nécessaires sont créés (unicité et recherche).
- ☐ Les **formats** de date/heure sont homogènes (DateOnly/TimeOnly si possible, sinon DateTime/TimeSpan).
- ☐ Les messages d'erreur utilisateur sont clairs (champ, contrainte, exemple attendu).

B. Spécialité (SPECIALTY)

- ☐ **Name** : string ≤ 30, Required, pas d'espaces-only.
- ☐ **Unicité** sur Name (ex. « Cardiologie » ne peut exister qu'une fois).
- ☐ Relation 1→N avec Médecin : on ne peut pas créer un Médecin avec une SpecialityId inexistante.
- ☐ Suppression d'une spécialité avec médecins associés : **interdite** (ou protégée) pour éviter des médecins orphelins.

C. Médecin (DOCTOR)

- ☐ **FirstName / LastName** : string ≤ 30, Required, sans blancs-only.
- ☐ **SpecialtyId** : Required, FK existante (référence valide à SPECIALTY).
- ☐ Index de recherche recommandé sur (LastName, FirstName).
- ☐ Un médecin peut avoir **0..N consultations** et **0..N ordonnances**.
- ☐ Suppression d'un médecin ayant consultations/ordonnances : **interdite** (historisation).

D. Patient (PATIENT)

- ☐ **FirstName / LastName** : string ≤ 30, Required, sans blancs-only.
- ☐ **BirthDate** : DateOnly (ou DateTime date), Required, **date plausible** (passé, ≥ 1900 par ex.).
- ☐ Option : **unicité logique** sur (LastName, FirstName, BirthDate) pour limiter les doublons.
- ☐ Un patient peut avoir **0..N consultations** et **0..N ordonnances**.
- ☐ Suppression d'un patient avec historique : **interdite** (traçabilité).

E. Consultation (CONSULTATION)

- ☐ **DoctorId** : Required, FK valide.
- ☐ **PatientId** : Required, FK valide.
- ☐ **Date** : DateOnly (ou DateTime date), Required.
- ☐ **Hour** : TimeOnly (ou TimeSpan), Required.
- ☐ **Reason** : string ≤ 100, Optional (peut être vide).
- ☐ Index **(DoctorId, Date, Hour)** pour détecter les collisions d'agenda.
- ☐ Règle métier : **pas de double-booking** pour un même médecin à la même date/heure.
- ☐ Suppression d'une consultation liée à des ordonnances : **interdite** (ou contrôlée) sauf si la FK ConsultationId est optionnelle et retirée proprement.

F. Médicament (MEDICAMENT)

- ☐ **Name** : string ≤ 100, Required, quasi-unique.
- ☐ **DosageForm** : string ≤ 30, Required (ex. comprimé, sirop…).
- ☐ **Strength** : string ≤ 30, Required (ex. 500 mg).
- ☐ **AtcCode** : string ≤ 20, Optional (si utilisé).
- ☐ Option d'unicité : (Name, DosageForm, Strength) pour éviter doublons identiques.

- ☐ Un médicament peut être présent dans **0..N lignes d'ordonnance**.
- ☐ Suppression d'un médicament référencé par des lignes : **interdite** (intégrité historique).

G. Ordonnance (ORDONNANCE)

- ☐ **DoctorId** : Required, FK valide.
- ☐ **PatientId** : Required, FK valide.
- ☐ **ConsultationId** : Optional, FK cohérente (si présente, le patient et le médecin correspondent).
- ☐ **Date** : DateOnly (ou DateTime date), Required (≥ date de la consultation, si liée).
- ☐ **Notes** : string ≤ 255, Optional.
- ☐ **Au moins 1 ligne** d'ordonnance associée (contrôle métier à la création).
- ☐ Suppression d'une ordonnance : **cascade** interdite vers patient/médecin ; **cascade autorisée** vers ses lignes.

H. Ligne d'ordonnance (ORDONNANCE_LIGNE)

- ☐ **OrdonnanceId** : Required, FK valide.
- ☐ **MedicamentId** : Required, FK valide.
- ☐ **Dosage** : string ≤ 50, Required (ex. « 500 mg »).
- ☐ **Frequency** : string ≤ 50, Required (ex. « 3 fois/jour »).
- ☐ **Duration** : string ≤ 30, Required (ex. « 7 jours » ou « au besoin »).
- ☐ **Quantity** : int > 0, Required (unité claire : boîtes, unités).
- ☐ **Instructions** : string ≤ 255, Optional.
- ☐ Option d'unicité intra-ordonnance : (OrdonnanceId, MedicamentId, Dosage, Frequency, Duration) pour éviter doublons exacts.
- ☐ Suppression d'une ordonnance doit **supprimer** ses lignes (cascade locale).

I. Intégrité et cohérence transverses

- ☐ Toute FK pointe vers un **enregistrement existant** (vérifiée avant insertion).
- ☐ Contrainte : l'**Ordonnance.PatientId** = **Consultation.PatientId** quand ConsultationId est renseigné.
- ☐ Contrainte : l'**Ordonnance.DoctorId** = **Consultation.DoctorId** quand ConsultationId est renseigné.
- ☐ Les dates/heures d'une consultation sont **logiques** (pas dans le passé lointain si non prévu, pas d'heure impossible).
- ☐ Les textes sont **trim**-és, pas de valeurs uniquement composées d'espaces.

J. Index & performances

- ☐ Index **unique** : Specialty.Name ; optionnellement Medicament.(Name, DosageForm, Strength).
- ☐ Index **recherche** : Doctor(LastName, FirstName) ; Patient(LastName, FirstName, BirthDate).
- ☐ Index **agenda** : Consultation(DoctorId, Date, Hour).
- ☐ Plans de test avec ~1k consultations et ~5k lignes d'ordonnance restent performants.

K. Validation côté ASP.NET Core (fonctionnelle)

- ☐ Les champs Required affichent des **messages d'erreur** explicites (FR).
- ☐ Les longueurs max déclenchent des erreurs lisibles (« 30 caractères max »).
- ☐ **Quantity > 0** contrôlé et expliqué.
- ☐ Messages guides pour date/heure (« format attendu », exemples).

L. Scénarios de test (à cocher)

Créations "happy path"

- ☐ Créer 4 spécialités (Cardio, Dermato, Neuro, Ophta).
- ☐ Créer 3 médecins rattachés à ces spécialités.
- ☐ Créer 5 patients valides avec BirthDate plausible.
- ☐ Créer 1 consultation par patient (date+heure valides, motif renseigné).
- ☐ Créer 1 catalogue de 5–10 médicaments (Name, DosageForm, Strength).
- ☐ Émettre 1 ordonnance liée à une consultation, avec 1–3 lignes conformes (dosage, fréquence, durée, quantité).

Erreurs attendues

- ☐ Tentative de créer un médecin avec SpecialtyId inexistant → **rejetée**.
- ☐ Double-booking même médecin/date/heure → **rejeté**.
- ☐ Ordonnance sans lignes → **rejetée**.
- ☐ Ligne avec Quantity ≤ 0 → **rejetée**.
- ☐ Consultation/Ordonnance avec FK cassée → **rejetée**.
- ☐ Incohérence Ordonnance vs Consultation (patient ou médecin différents) → **rejetée**.
- ☐ Spécialité en double (Name identique) → **rejetée**.
- ☐ Médicament strictement dupliqué (si unicité activée) → **rejetée**.

Suppression / intégrité

- ☐ Suppression d'une spécialité ayant des médecins → **interdite**.
- ☐ Suppression d'un patient/medecin ayant historique → **interdite**.
- ☐ Suppression d'une ordonnance → supprime **uniquement** ses lignes.

Temps de travail :

```
3 ects = 3 x 30h = 90h
15h + 15h = 30h
2h + 1h + 8h = 11h
90h - 41h = 49h
Entités      : 3h
DTO          : 3h
Mapper       : 1h
Projet       : 1h
Repository   : 3h
Services     : 3h
Contrôleurs  : 3h
Postman      : 5h
             : 22h * 1.5 = 33h
```