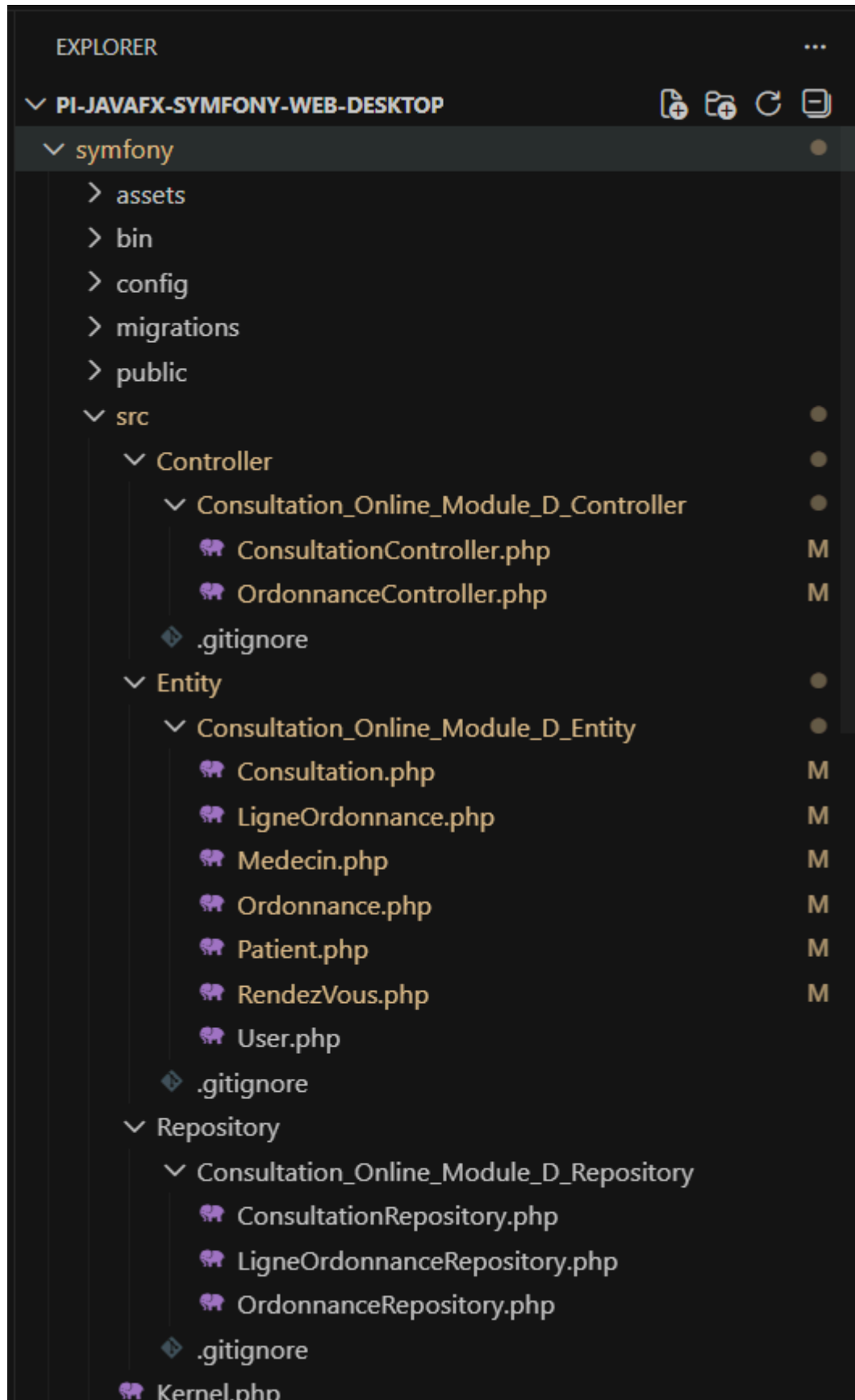
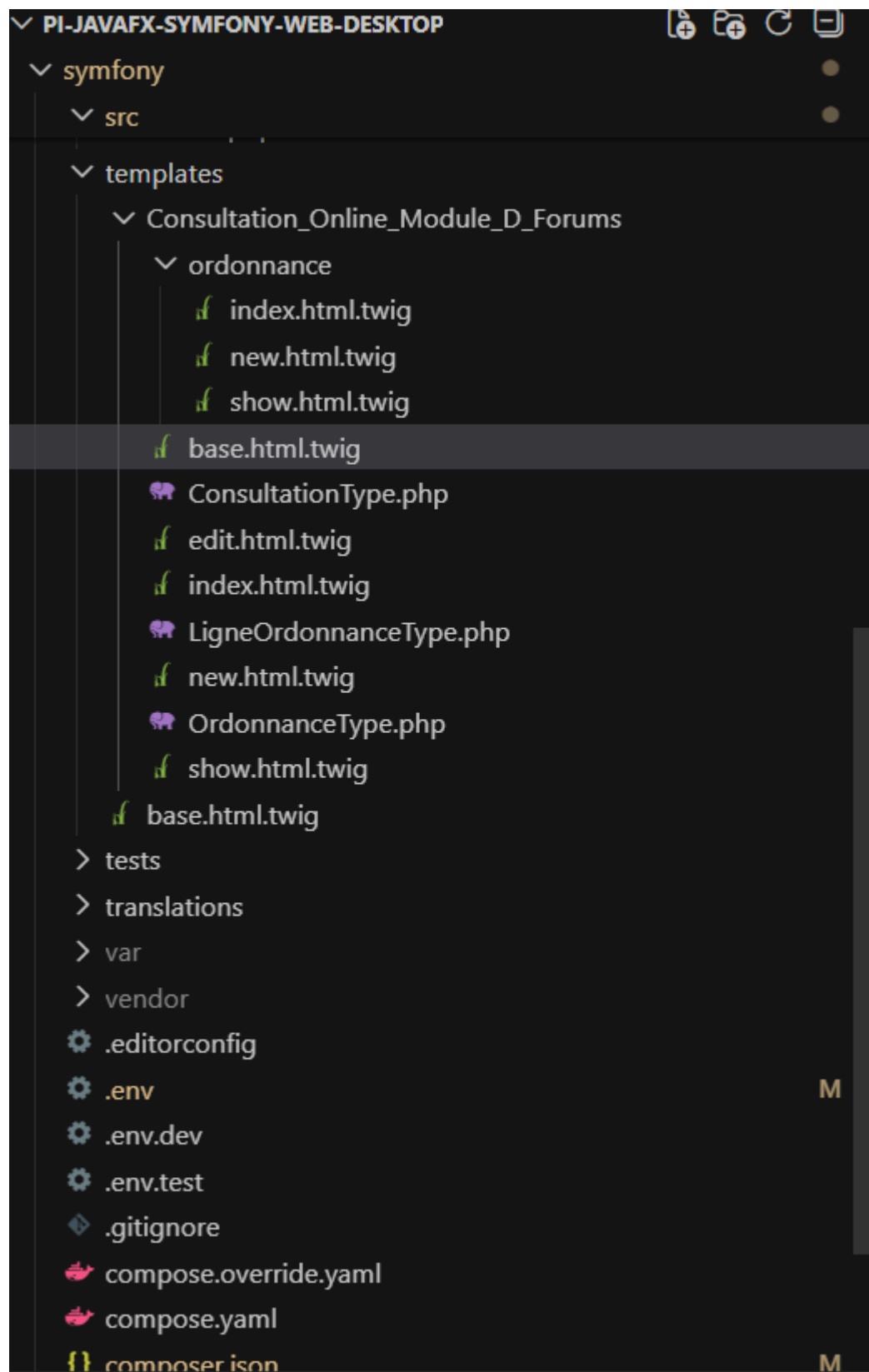


voici l'architecture de mon module dans le projet de MediSmart





```

{} composer.json M
{} composer.lock M
🐘 importmap.php
🔗 phpunit.dist.xml
≡ symfony.lock

```

D:\PI-JAVAFX-SYMFONY-WEB-DESKTOP\symfony\src\Controller\Consultation_Online_Module_D_Controller

<?php

```

namespace App\Consultation_Online_Module_D_Entity\Controller;

use App\Consultation_Online_Module_D_Entity\Entity\Consultation;
use App\Consultation_Online_Module_D_Forums\Form\ConsultationType;
use
App\Consultation_Online_Module_D_Repository\Repository\ConsultationRepository;
use Doctrine\ORM\EntityManagerInterface;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;
use Symfony\Component\Validator\Validator\ValidatorInterface;

#[Route('/consultation')]
class ConsultationController extends AbstractController
{
    #[Route('/consultation', name: 'app_consultation_index', methods: ['GET'])]
    public function index(Request $request, ConsultationRepository $consultationRepository): Response
    {
        // Récupérer les paramètres de recherche et tri
        $search = $request->query->get('search', '');
        $sort = $request->query->get('sort', 'dateDebut');
        $order = $request->query->get('order', 'DESC');

        $queryBuilder =
        $consultationRepository->createQueryBuilder('c')
            ->leftJoin('c.patient', 'p')

```

```

        ->leftJoin('c.medecin', 'm');

// FONCTIONNALITÉ AVANCÉE : Recherche
if ($search) {
    $queryBuilder
        ->where('p.nom LIKE :search OR p.prenom LIKE :search')
        ->orWhere('m.nom LIKE :search OR m.prenom LIKE
:search')

        ->orWhere('c.diagnostic LIKE :search')
        ->orWhere('c.motif LIKE :search')
        ->setParameter('search', '%' . $search . '%');
}

// FONCTIONNALITÉ AVANCÉE : Tri
$validSortFields = ['dateDebut', 'statut', 'type', 'id'];
if (in_array($sort, $validSortFields)) {
    $queryBuilder->orderBy('c.' . $sort, $order === 'ASC' ?
'ASC' : 'DESC');
}

$consultations = $queryBuilder->getQuery()->getResult();

return $this->render('consultation/index.html.twig', [
    'consultations' => $consultations,
    'search' => $search,
    'sort' => $sort,
    'order' => $order,
]);
}

#[Route('/new', name: 'app_consultation_new', methods: ['GET',
'POST'])]
public function new(Request $request, EntityManagerInterface
$entityManager, ValidatorInterface $validator): Response
{
    $consultation = new Consultation();
    $form = $this->createForm(ConsultationType::class,
$consultation);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {

        // VALIDATION CÔTÉ SERVEUR (pas de JavaScript!)

```

```

        $errors = $validator->validate($consultation);

        if (count($errors) > 0) {
            foreach ($errors as $error) {
                $this->addFlash('error', $error->getMessage());
            }
            return $this->render('consultation/new.html.twig', [
                'consultation' => $consultation,
                'form' => $form,
            ]);
        }

        // RÈGLES MÉTIER PERSONNALISÉES
        if ($consultation->getDateDebut() < new \DateTime()) {
            $this->addFlash('error', 'La date de consultation ne
peut pas être dans le passé');
            return $this->render('consultation/new.html.twig', [
                'consultation' => $consultation,
                'form' => $form,
            ]);
        }

        // Initialiser les valeurs par défaut
        $consultation->setStatut('PLANIFIEE');
        $consultation->setPayee(false);

        $entityManager->persist($consultation);
        $entityManager->flush();

        $this->addFlash('success', 'Consultation créée avec succès
!');

        return $this->redirectToRoute('app_consultation_index');
    }

    return $this->render('consultation/new.html.twig', [
        'consultation' => $consultation,
        'form' => $form,
    ]);
}

#[Route('/{id}', name: 'app_consultation_show', methods: ['GET'])]
public function show(Consultation $consultation): Response
{

```

```

        return $this->render('consultation/show.html.twig', [
            'consultation' => $consultation,
        ]);
    }

    #[Route('/{id}/edit', name: 'app_consultation_edit', methods:
['GET', 'POST'])]
    public function edit(Request $request, Consultation $consultation,
EntityManagerInterface $entityManager, ValidatorInterface $validator):
Response
    {
        $form = $this->createForm(ConsultationType::class,
$consultation);
        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {

            // VALIDATION SERVEUR
            $errors = $validator->validate($consultation);

            if (count($errors) > 0) {
                foreach ($errors as $error) {
                    $this->addFlash('error', $error->getMessage());
                }
                return $this->render('consultation/edit.html.twig', [
                    'consultation' => $consultation,
                    'form' => $form,
                ]);
            }

            $entityManager->flush();

            $this->addFlash('success', 'Consultation modifiée avec
succès !');
            return $this->redirectToRoute('app_consultation_show',
['id' => $consultation->getId()]);
        }

        return $this->render('consultation/edit.html.twig', [
            'consultation' => $consultation,
            'form' => $form,
        ]);
    }

```

```

        #[Route('/{id}', name: 'app_consultation_delete', methods:
['POST'])]
        public function delete(Request $request, Consultation
$consultation, EntityManagerInterface $entityManager): Response
        {
            if ($this->isCsrfTokenValid('delete'.$consultation->getId(),
$request->request->get('_token'))) {
                $entityManager->remove($consultation);
                $entityManager->flush();
                $this->addFlash('success', 'Consultation supprimée avec
succès !');
            }

            return $this->redirectToRoute('app_consultation_index');
        }

        // Actions supplémentaires
        #[Route('/{id}/demarrer', name: 'app_consultation_demarrer',
methods: ['POST'])]
        public function demarrer(Consultation $consultation,
EntityManagerInterface $entityManager): Response
        {
            $consultation->setStatut('EN_COURS');
            $consultation->setDateDebut(new \DateTime());
            $entityManager->flush();

            $this->addFlash('success', 'Consultation démarrée');
            return $this->redirectToRoute('app_consultation_show', ['id' =>
$consultation->getId()]);
        }

        #[Route('/{id}/terminer', name: 'app_consultation_terminer',
methods: ['POST'])]
        public function terminer(Consultation $consultation,
EntityManagerInterface $entityManager): Response
        {
            $consultation->setStatut('TERMINEE');
            $consultation->setDateFin(new \DateTime());

            // Calculer durée
            if ($consultation->getDateDebut()) {

```

```

        $duree =
$consultation->getDateDebut()->diff($consultation->getDateFin());
        $consultation->setDureeMinutes($duree->i + ($duree->h *
60));
    }

    $entityManager->flush();

    $this->addFlash('success', 'Consultation terminée');
    return $this->redirectToRoute('app_consultation_show', ['id' =>
$consultation->getId()]);
    }
}
<?php
// src/Controller/OrdonnanceController.php

namespace App\Consultation_Online_Module_D_Controller\Controller;

use App\Consultation_Online_Module_D_Entity\Entity\Ordonnance;
use App\Consultation_Online_Module_D_Forums\Form\OrdonnanceType;
use
App\Consultation_Online_Module_D_Repository\Repository\OrdonnanceReposi
tory;
use Doctrine\ORM\EntityManagerInterface;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

use Symfony\Component\Validator\Validator\ValidatorInterface;

#[Route('/ordonnance')]
class OrdonnanceController extends AbstractController
{
    #[Route('/', name: 'app_ordonnance_index', methods: ['GET'])]
    public function index(OrdonnanceRepository $ordonnanceRepository):
Response
    {
        return $this->render('ordonnance/index.html.twig', [
            'ordonnances' => $ordonnanceRepository->findAll(),
        ]);
    }
}

```



```

    #[Route('/new', name: 'app_ordonnance_new', methods: ['GET',
'POST'])]

    public function new(Request $request, EntityManagerInterface
$entityManager, ValidatorInterface $validator): Response
    {
        $ordonnance = new Ordonnance();
        $form = $this->createForm(OrdonnanceType::class, $ordonnance);
        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {

            // VALIDATION SERVEUR
            $errors = $validator->validate($ordonnance);

            if (count($errors) > 0) {
                foreach ($errors as $error) {
                    $this->addFlash('error', $error->getMessage());
                }
                return $this->render('ordonnance/new.html.twig', [
                    'ordonnance' => $ordonnance,
                    'form' => $form,
                ]);
            }

            // Vérifier qu'il y a au moins un médicament
            if ($ordonnance->getLignes()->count() === 0) {
                $this->addFlash('error', 'Vous devez ajouter au moins
un médicament');
                return $this->render('ordonnance/new.html.twig', [
                    'ordonnance' => $ordonnance,
                    'form' => $form,
                ]);
            }

            $entityManager->persist($ordonnance);
            $entityManager->flush();

            $this->addFlash('success', 'Ordonnance créée avec succès
!');

            return $this->redirectToRoute('app_ordonnance_index');
        }
    }

```

```

        return $this->render('ordonnance/new.html.twig', [
            'ordonnance' => $ordonnance,
            'form' => $form,
        ]);
    }

    #[Route('/{id}', name: 'app_ordonnance_show', methods: ['GET'])]
    public function show(Ordonnance $ordonnance): Response
    {
        return $this->render('ordonnance/show.html.twig', [
            'ordonnance' => $ordonnance,
        ]);
    }

    #[Route('/{id}/edit', name: 'app_ordonnance_edit', methods: ['GET',
'POST'])]
    public function edit(Request $request, Ordonnance $ordonnance,
EntityManagerInterface $entityManager): Response
    {
        $form = $this->createForm(OrdonnanceType::class, $ordonnance);
        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {
            $entityManager->flush();

            $this->addFlash('success', 'Ordonnance modifiée avec succès
!');

            return $this->redirectToRoute('app_ordonnance_show', ['id'
=> $ordonnance->getId()]);
        }

        return $this->render('ordonnance/edit.html.twig', [
            'ordonnance' => $ordonnance,
            'form' => $form,
        ]);
    }

    #[Route('/{id}', name: 'app_ordonnance_delete', methods: ['POST'])]
    public function delete(Request $request, Ordonnance $ordonnance,
EntityManagerInterface $entityManager): Response
    {
        if ($this->isCsrfTokenValid('delete'.$ordonnance->getId(),
$request->request->get('_token'))) {

```

```

        $entityManager->remove($ordonnance);
        $entityManager->flush();
        $this->addFlash('success', 'Ordonnance supprimée');
    }

    return $this->redirectToRoute('app_ordonnance_index');
}
}

```

D:\PI-JAVAFX-SYMFONY-WEB-DESKTOP\symfony\src\Entity\Consultation_Online_Module_D_Entity

<?php

```

namespace App\Consultation_Online_Module_D_Entity\Entity;
use
App\Consultation_Online_Module_D_Repository\Repository\ConsultationRepository;
use Doctrine\DBAL\Types\Types;
use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Validator\Constraints as Assert;
use App\Consultation_Online_Module_D_Entity\Entity\Patient;
use App\Consultation_Online_Module_D_Entity\Entity\Medecin;
use App\Consultation_Online_Module_D_Entity\Entity\RendezVous;
use App\Consultation_Online_Module_D_Entity\Entity\Ordonnance;

#[ORM\Entity(repositoryClass: ConsultationRepository::class)]
class Consultation
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\ManyToOne(inversedBy: 'consultations')]
    #[ORM\JoinColumn(nullable: false)]
    #[Assert\NotNull(message: "Le patient est obligatoire")]
    private ?Patient $patient = null;

    #[ORM\ManyToOne(inversedBy: 'consultations')]
    #[ORM\JoinColumn(nullable: false)]
    #[Assert\NotNull(message: "Le médecin est obligatoire")]
    private ?Medecin $medecin = null;
}

```

```

#[ORM\ManyToOne]
private ?RendezVous $rendezVous = null;

#[ORM\Column(type: Types::DATETIME_MUTABLE)]
#[Assert\NotBlank(message: "La date de début est obligatoire")]
private ?\DateTimeInterface $dateDebut = null;

#[ORM\Column(type: Types::DATETIME_MUTABLE, nullable: true)]
private ?\DateTimeInterface $dateFin = null;

#[ORM\Column(nullable: true)]
private ?int $dureeMinutes = null;

#[ORM\Column(length: 50)]
#[Assert\NotBlank(message: "Le type est obligatoire")]
#[Assert\Choice(
    choices: ['VIDEO', 'CABINET', 'AUDIO'],
    message: "Type invalide. Choisir VIDEO, CABINET ou AUDIO"
)]
private ?string $type = 'VIDEO';

#[ORM\Column(length: 50)]
#[Assert\Choice(
    choices: ['PLANIFIEE', 'EN_ATTENTE', 'EN_COURS', 'TERMINEE',
'ANNULEE', 'NO_SHOW'],
    message: "Statut invalide"
)]
private ?string $statut = 'PLANIFIEE';

#[ORM\Column(type: Types::TEXT, nullable: true)]
private ?string $motif = null;

#[ORM\Column(type: Types::TEXT, nullable: true)]
#[Assert\Length(max: 2000, maxMessage: "Le diagnostic ne peut pas
dépasser {{ limit }} caractères")]
private ?string $diagnostic = null;

#[ORM\Column(type: Types::TEXT, nullable: true)]
private ?string $notesConsultation = null;

#[ORM\Column(type: Types::TEXT, nullable: true)]
private ?string $symptomes = null;

```

```

#[ORM\Column(length: 255, nullable: true)]
private ?string $urlVisio = null;

#[ORM\Column(length: 255, nullable: true)]
private ?string $tokenSession = null;

#[ORM\Column(nullable: true)]
#[Assert\Range(min: 1, max: 5, notInRangeMessage: "La satisfaction
doit être entre {{ min }} et {{ max }}")]
private ?int $satisfaction = null;

#[ORM\Column(type: Types::DECIMAL, precision: 10, scale: 2,
nullable: true)]
#[Assert\Positive(message: "Le montant doit être positif")]
private ?string $montant = null;

#[ORM\Column]
private ?bool $payee = false;

#[ORM\OneToOne(mappedBy: 'consultation', cascade: ['persist',
'remove'])]
private ?Ordonnance $ordonnance = null;

public function getId(): ?int
{
    return $this->id;
}

public function getPatient(): ?Patient
{
    return $this->patient;
}

public function setPatient(?Patient $patient): static
{
    $this->patient = $patient;
    return $this;
}

public function getMedecin(): ?Medecin
{
    return $this->medecin;
}

```

```
public function setMedecin(?Medecin $medecin): static
{
    $this->medecin = $medecin;
    return $this;
}

public function getRendezVous(): ?RendezVous
{
    return $this->rendezVous;
}

public function setRendezVous(?RendezVous $rendezVous): static
{
    $this->rendezVous = $rendezVous;
    return $this;
}

public function getDateDebut(): ?\DateTimeInterface
{
    return $this->dateDebut;
}

public function setDateDebut(\DateTimeInterface $dateDebut): static
{
    $this->dateDebut = $dateDebut;
    return $this;
}

public function getDateFin(): ?\DateTimeInterface
{
    return $this->dateFin;
}

public function setDateFin(?\DateTimeInterface $dateFin): static
{
    $this->dateFin = $dateFin;
    return $this;
}

public function getDureeMinutes(): ?int
{
    return $this->dureeMinutes;
}
```

```
}

public function setDureeMinutes(?int $dureeMinutes): static
{
    $this->dureeMinutes = $dureeMinutes;
    return $this;
}

public function getType(): ?string
{
    return $this->type;
}

public function setType(string $type): static
{
    $this->type = $type;
    return $this;
}

public function getStatut(): ?string
{
    return $this->statut;
}

public function setStatut(string $statut): static
{
    $this->statut = $statut;
    return $this;
}

public function getMotif(): ?string
{
    return $this->motif;
}

public function setMotif(?string $motif): static
{
    $this->motif = $motif;
    return $this;
}

public function getDiagnostic(): ?string
{

```

```
        return $this->diagnostic;
    }

    public function setDiagnostic(?string $diagnostic): static
    {
        $this->diagnostic = $diagnostic;
        return $this;
    }

    public function getNotesConsultation(): ?string
    {
        return $this->notesConsultation;
    }

    public function setNotesConsultation(?string $notesConsultation):
static
    {
        $this->notesConsultation = $notesConsultation;
        return $this;
    }

    public function getSymptomes(): ?string
    {
        return $this->symptomes;
    }

    public function setSymptomes(?string $symptomes): static
    {
        $this->symptomes = $symptomes;
        return $this;
    }

    public function getUrlVisio(): ?string
    {
        return $this->urlVisio;
    }

    public function setUrlVisio(?string $urlVisio): static
    {
        $this->urlVisio = $urlVisio;
        return $this;
    }
}
```



```
public function getTokenSession(): ?string
{
    return $this->tokenSession;
}

public function setTokenSession(?string $tokenSession): static
{
    $this->tokenSession = $tokenSession;
    return $this;
}

public function getSatisfaction(): ?int
{
    return $this->satisfaction;
}

public function setSatisfaction(?int $satisfaction): static
{
    $this->satisfaction = $satisfaction;
    return $this;
}

public function getMontant(): ?string
{
    return $this->montant;
}

public function setMontant(?string $montant): static
{
    $this->montant = $montant;
    return $this;
}

public function isPayee(): ?bool
{
    return $this->payee;
}

public function setPayee(bool $payee): static
{
    $this->payee = $payee;
    return $this;
}
```

```

    public function getOrdonnance(): ?Ordonnance
    {
        return $this->ordonnance;
    }

    public function setOrdonnance(?Ordonnance $ordonnance): static
    {
        if ($ordonnance === null && $this->ordonnance !== null) {
            $this->ordonnance->setConsultation(null);
        }

        if ($ordonnance !== null && $ordonnance->getConsultation() !==
$this) {
            $ordonnance->setConsultation($this);
        }

        $this->ordonnance = $ordonnance;
        return $this;
    }
}
<?php
// src/Entity/LigneOrdonnance.php

namespace App\Consultation_Online_Module_D_Entity\Entity;
use
App\Consultation_Online_Module_D_Repository\Repository\LigneOrdonnanceR
epository;
use Doctrine\DBAL\Types\Types;
use Doctrine\ORM\Mapping as ORM;
use App\Consultation_Online_Module_D_Entity\Entity\Ordonnance;

use Symfony\Component\Validator\Constraints as Assert;

#[ORM\Entity(repositoryClass: LigneOrdonnanceRepository::class)]
class LigneOrdonnance
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\ManyToOne(inversedBy: 'lignes')]

```

```

#[ORM\JoinColumn(nullable: false)]
private ?Ordonnance $ordonnance = null;

#[ORM\Column(length: 255)]
#[Assert\NotBlank(message: "Le nom du médicament est obligatoire")]
#[Assert\Length(
    min: 3,
    max: 255,
    minMessage: "Le nom doit faire au moins {{ limit }}
caractères",
    maxMessage: "Le nom ne peut pas dépasser {{ limit }}
caractères"
)]
private ?string $nomMedicament = null;

#[ORM\Column(length: 100)]
#[Assert\NotBlank(message: "Le dosage est obligatoire")]
private ?string $dosage = null;

#[ORM\Column(length: 50)]
#[Assert\Choice(
    choices: ['COMPRIME', 'GELULE', 'SIROP', 'AMPOULE', 'POMMADE',
'CREME', 'GOUTTES', 'SUPPOSITOIRE'],
    message: "Forme invalide"
)]
private ?string $forme = 'COMPRIME';

#[ORM\Column]
#[Assert\Positive(message: "La quantité doit être positive")]
#[Assert\Range(
    min: 1,
    max: 999,
    notInRangeMessage: "La quantité doit être entre {{ min }} et {{
max }}"
)]
private ?int $quantite = null;

#[ORM\Column(length: 255)]
#[Assert\NotBlank(message: "La fréquence est obligatoire")]
private ?string $frequence = null;

#[ORM\Column]

```

```
#[Assert\Positive(message: "La durée du traitement doit être
positive")]
private ?int $dureeTraitement = null;

#[ORM\Column(type: Types::TEXT, nullable: true)]
private ?string $instructions = null;

public function getId(): ?int
{
    return $this->id;
}

public function getOrdonnance(): ?Ordonnance
{
    return $this->ordonnance;
}

public function setOrdonnance(?Ordonnance $ordonnance): static
{
    $this->ordonnance = $ordonnance;
    return $this;
}

public function getNomMedicament(): ?string
{
    return $this->nomMedicament;
}

public function setNomMedicament(string $nomMedicament): static
{
    $this->nomMedicament = $nomMedicament;
    return $this;
}

public function getDosage(): ?string
{
    return $this->dosage;
}

public function setDosage(string $dosage): static
{
    $this->dosage = $dosage;
    return $this;
}
```

```
}

public function getForme(): ?string
{
    return $this->forme;
}

public function setForme(string $forme): static
{
    $this->forme = $forme;
    return $this;
}

public function getQuantite(): ?int
{
    return $this->quantite;
}

public function setQuantite(int $quantite): static
{
    $this->quantite = $quantite;
    return $this;
}

public function getFrequence(): ?string
{
    return $this->frequence;
}

public function setFrequence(string $frequence): static
{
    $this->frequence = $frequence;
    return $this;
}

public function getDureeTraitement(): ?int
{
    return $this->dureeTraitement;
}

public function setDureeTraitement(int $dureeTraitement): static
{
    $this->dureeTraitement = $dureeTraitement;
}
```

```

        return $this;
    }

    public function getInstructions(): ?string
    {
        return $this->instructions;
    }

    public function setInstructions(?string $instructions): static
    {
        $this->instructions = $instructions;
        return $this;
    }
}
<?php
// src/Entity/Medecin.php
namespace App\Consultation_Online_Module_D_Entity\Entity;

use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\ORM\Mapping as ORM;
use App\Consultation_Online_Module_D_Entity\Entity\Consultation;

#[ORM\Entity]
class Medecin
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column(length: 100)]
    private ?string $nom = null;

    #[ORM\Column(length: 100)]
    private ?string $prenom = null;

    #[ORM\Column(length: 100)]
    private ?string $specialite = null;

```

```

#[ORM\Column(length: 180, unique: true)]
private ?string $email = null;

#[ORM\OneToMany(targetEntity: Consultation::class, mappedBy:
'medecin')]
private Collection $consultations;

public function __construct()
{
    $this->consultations = new ArrayCollection();
}

// Getters/Setters
public function getId(): ?int { return $this->id; }
public function getNom(): ?string { return $this->nom; }
public function setNom(string $nom): self { $this->nom = $nom;
return $this; }
    public function getPrenom(): ?string { return $this->prenom; }
    public function setPrenom(string $prenom): self { $this->prenom =
$prenom; return $this; }
    public function getSpecialite(): ?string { return
$this->specialite; }
    public function setSpecialite(string $specialite): self {
$this->specialite = $specialite; return $this; }
    public function getEmail(): ?string { return $this->email; }
    public function setEmail(string $email): self { $this->email =
$email; return $this; }
    public function getConsultations(): Collection { return
$this->consultations; }

public function __toString(): string
{
    return 'Dr. ' . $this->nom . ' - ' . $this->specialite;
}
}
<?php
// src/Entity/Ordonnance.php

namespace App\Consultation_Online_Module_D_Entity\Entity;

```

```

use
App\Consultation_Online_Module_D_Repository\Repository\OrdonnanceRepository;
use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\DBAL\Types\Types;
use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Validator\Constraints as Assert;

use App\Consultation_Online_Module_D_Entity\Entity\Consultation;
use App\Consultation_Online_Module_D_Entity\Entity\Patient;
use App\Consultation_Online_Module_D_Entity\Entity\Medecin;
use App\Consultation_Online_Module_D_Entity\Entity\LigneOrdonnance;

#[ORM\Entity(repositoryClass: OrdonnanceRepository::class)]
class Ordonnance
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column(length: 100, unique: true)]
    private ?string $numeroOrdonnance = null;

    #[ORM\OneToOne(inversedBy: 'ordonnance', cascade: ['persist', 'remove'])]
    #[ORM\JoinColumn(nullable: false)]
    #[Assert\NotNull(message: "La consultation est obligatoire")]
    private ?Consultation $consultation = null;

    #[ORM\ManyToOne]
    #[ORM\JoinColumn(nullable: false)]
    #[Assert\NotNull(message: "Le patient est obligatoire")]
    private ?Patient $patient = null;

    #[ORM\ManyToOne]
    #[ORM\JoinColumn(nullable: false)]
    #[Assert\NotNull(message: "Le médecin est obligatoire")]
    private ?Medecin $medecin = null;

```



```

#[ORM\Column(type: Types::DATE_MUTABLE)]
#[Assert\NotBlank(message: "La date d'émission est obligatoire")]
private ?\DateTimeInterface $dateEmission = null;

#[ORM\Column(type: Types::DATE_MUTABLE)]
#[Assert\NotBlank(message: "La date de validité est obligatoire")]
#[Assert\GreaterThan(propertyPath: "dateEmission", message: "La
date de validité doit être après la date d'émission")]
private ?\DateTimeInterface $dateValidite = null;

#[ORM\Column(type: Types::TEXT, nullable: true)]
private ?string $diagnostic = null;

#[ORM\Column(type: Types::TEXT, nullable: true)]
private ?string $instructions = null;

#[ORM\Column]
private ?bool $delivree = false;

#[ORM\Column]
private ?bool $utilisee = false;

#[ORM\Column(length: 255, nullable: true)]
private ?string $qrCode = null;

#[ORM\Column(length: 255, nullable: true)]
private ?string $pdfPath = null;

/**
 * @var Collection<int, LigneOrdonnance>
 */
#[ORM\OneToMany(targetEntity: LigneOrdonnance::class, mappedBy:
'ordonnance', cascade: ['persist', 'remove'], orphanRemoval: true)]
#[Assert\Count(min: 1, minMessage: "Vous devez ajouter au moins un
médicament")]
private Collection $lignes;

public function __construct()
{
    $this->lignes = new ArrayCollection();
    $this->numeroOrdonnance = 'ORD-' . date('Ymd') . '-' .
uniqid();
    $this->dateEmission = new \DateTime();

```

```

        $this->dateValidite = new \DateTime('+1 month');
    }

    public function getId(): ?int
    {
        return $this->id;
    }

    public function getNumeroOrdonnance(): ?string
    {
        return $this->numeroOrdonnance;
    }

    public function setNumeroOrdonnance(string $numeroOrdonnance):
static
    {
        $this->numeroOrdonnance = $numeroOrdonnance;
        return $this;
    }

    public function getConsultation(): ?Consultation
    {
        return $this->consultation;
    }

    public function setConsultation(?Consultation $consultation):
static
    {
        $this->consultation = $consultation;
        return $this;
    }

    public function getPatient(): ?Patient
    {
        return $this->patient;
    }

    public function setPatient(?Patient $patient): static
    {
        $this->patient = $patient;
        return $this;
    }

```

```
public function getMedecin(): ?Medecin
{
    return $this->medecin;
}

public function setMedecin(?Medecin $medecin): static
{
    $this->medecin = $medecin;
    return $this;
}

public function getDateEmission(): ?\DateTimeInterface
{
    return $this->dateEmission;
}

public function setDateEmission(\DateTimeInterface $dateEmission):
static
{
    $this->dateEmission = $dateEmission;
    return $this;
}

public function getDateValidite(): ?\DateTimeInterface
{
    return $this->dateValidite;
}

public function setDateValidite(\DateTimeInterface $dateValidite):
static
{
    $this->dateValidite = $dateValidite;
    return $this;
}

public function getDiagnostic(): ?string
{
    return $this->diagnostic;
}

public function setDiagnostic(?string $diagnostic): static
{
    $this->diagnostic = $diagnostic;
}
```

```
        return $this;
    }

    public function getInstructions(): ?string
    {
        return $this->instructions;
    }

    public function setInstructions(?string $instructions): static
    {
        $this->instructions = $instructions;
        return $this;
    }

    public function isDelivree(): ?bool
    {
        return $this->delivree;
    }

    public function setDelivree(bool $delivree): static
    {
        $this->delivree = $delivree;
        return $this;
    }

    public function isUtilisee(): ?bool
    {
        return $this->utilisee;
    }

    public function setUtilisee(bool $utilisee): static
    {
        $this->utilisee = $utilisee;
        return $this;
    }

    public function getQrCode(): ?string
    {
        return $this->qrCode;
    }

    public function setQrCode(?string $qrCode): static
    {

```

```

        $this->qrCode = $qrCode;
        return $this;
    }

    public function getPdfPath(): ?string
    {
        return $this->pdfPath;
    }

    public function setPdfPath(?string $pdfPath): static
    {
        $this->pdfPath = $pdfPath;
        return $this;
    }

    /**
     * @return Collection<int, LigneOrdonnance>
     */
    public function getLignes(): Collection
    {
        return $this->lignes;
    }

    public function addLigne(LigneOrdonnance $ligne): static
    {
        if (!$this->lignes->contains($ligne)) {
            $this->lignes->add($ligne);
            $ligne->setOrdonnance($this);
        }

        return $this;
    }

    public function removeLigne(LigneOrdonnance $ligne): static
    {
        if ($this->lignes->removeElement($ligne)) {
            if ($ligne->getOrdonnance() === $this) {
                $ligne->setOrdonnance(null);
            }
        }

        return $this;
    }

```

```

}
<?php
// src/Entity/Patient.php
namespace App\Consultation_Online_Module_D_Entity\Entity;

use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\ORM\Mapping as ORM;
use App\Consultation_Online_Module_D_Entity\Entity\Consultation;

#[ORM\Entity]
class Patient
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column(length: 100)]
    private ?string $nom = null;

    #[ORM\Column(length: 100)]
    private ?string $prenom = null;

    #[ORM\Column(length: 180, unique: true)]
    private ?string $email = null;

    #[ORM\Column(length: 20, nullable: true)]
    private ?string $telephone = null;

    #[ORM\OneToMany(targetEntity: Consultation::class, mappedBy:
'patient')]
    private Collection $consultations;

    public function __construct()
    {
        $this->consultations = new ArrayCollection();
    }

    // Getters/Setters basiques
    public function getId(): ?int { return $this->id; }

```

```

        public function getNom(): ?string { return $this->nom; }
        public function setNom(string $nom): self { $this->nom = $nom;
return $this; }

        public function getPrenom(): ?string { return $this->prenom; }
        public function setPrenom(string $prenom): self { $this->prenom =
$prenom; return $this; }

        public function getEmail(): ?string { return $this->email; }
        public function setEmail(string $email): self { $this->email =
$email; return $this; }

        public function getTelephone(): ?string { return $this->telephone;
}

        public function setTelephone(?string $telephone): self {
$this->telephone = $telephone; return $this; }

        public function getConsultations(): Collection { return
$this->consultations; }


        public function __toString(): string
        {
            return $this->prenom . ' ' . $this->nom;
        }
    }
}
<?php
// src/Entity/RendezVous.php
namespace App\Consultation_Online_Module_D_Entity\Entity;
use Doctrine\ORM\Mapping as ORM;

#[ORM\Entity]
class RendezVous
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column(type: 'datetime')]
    private ?\DateTimeInterface $dateHeure = null;

    #[ORM\Column(length: 50)]
    private ?string $statut = 'EN_ATTENTE';

    // Getters/Setters basiques
    public function getId(): ?int { return $this->id; }

```

```

    public function getDateHeure(): ?\DateTimeInterface { return
$this->dateHeure; }

    public function setDateHeure(\DateTimeInterface $dateHeure): self {
$this->dateHeure = $dateHeure; return $this; }

    public function getStatut(): ?string { return $this->statut; }

    public function setStatut(string $statut): self { $this->statut =
$statut; return $this; }
}

```

D:\PI-JAVAFX-SYMFONY-WEB-DESKTOP\symfony\src\Repository\Consultation_Online_Module_D_Repository

```

<?php
// src/Repository/ConsultationRepository.php

namespace App\Consultation_Online_Module_D_Repository\Repository;

use App\Consultation_Online_Module_D_Entity\Entity\Consultation;
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
use Doctrine\Persistence\ManagerRegistry;

class ConsultationRepository extends ServiceEntityRepository
{
    public function __construct(ManagerRegistry $registry)
    {
        parent::__construct($registry, Consultation::class);
    }

    public function save(Consultation $entity, bool $flush = false):
void
    {
        $this->getEntityManager()->persist($entity);

        if ($flush) {
            $this->getEntityManager()->flush();
        }
    }

    public function remove(Consultation $entity, bool $flush = false):
void
    {
        $this->getEntityManager()->remove($entity);

        if ($flush) {

```



```

        $this->getEntityManager()->flush();
    }
}
}
<?php
// src/Repository/LigneOrdonnanceRepository.php

namespace App\Consultation_Online_Module_D_Repository\Repository;

use App\Consultation_Online_Module_D_Entity\Entity\LigneOrdonnance;
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
use Doctrine\Persistence\ManagerRegistry;

class LigneOrdonnanceRepository extends ServiceEntityRepository
{
    public function __construct(ManagerRegistry $registry)
    {
        parent::__construct($registry, LigneOrdonnance::class);
    }
}
<?php
// src/Repository/OrdonnanceRepository.php

namespace App\Consultation_Online_Module_D_Repository\Repository;

use App\Consultation_Online_Module_D_Entity\Entity\Ordonnance;
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
use Doctrine\Persistence\ManagerRegistry;

class OrdonnanceRepository extends ServiceEntityRepository
{
    public function __construct(ManagerRegistry $registry)
    {
        parent::__construct($registry, Ordonnance::class);
    }
}

```

D:\PI-JAVAFX-SYMFONY-WEB-DESKTOP\symfony\templates\Consultation_Online_Module_D_Forums

```

<?php
// src/Form/ConsultationType.php

```

```

namespace App\Consultation_Online_Module_D_Forum\Form;

use App\Consultation_Online_Module_D_Entity\Entity\Consultation;
use App\Consultation_Online_Module_D_Entity\Entity\Patient;
use App\Consultation_Online_Module_D_Entity\Entity\Medecin;
use App\Consultation_Online_Module_D_Entity\Entity\RendezVous;
use Symfony\Bridge\Doctrine\Form\Type\EntityType;
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\Extension\Core\Type\ChoiceType;
use Symfony\Component\Form\Extension\Core\Type\DateTimeType;
use Symfony\Component\Form\Extension\Core\Type\TextareaType;
use Symfony\Component\Form\Extension\Core\Type\NumberType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;

class ConsultationType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array
$options): void
    {
        $builder
            ->add('patient', EntityType::class, [
                'class' => Patient::class,
                'choice_label' => function(Patient $patient) {
                    return $patient->getPrenom() . ' ' .
                    $patient->getNom();
                },
                'placeholder' => '-- Sélectionner un patient --',
                'label' => 'Patient *',
                'attr' => ['class' => 'form-control'],
            ])
            ->add('medecin', EntityType::class, [
                'class' => Medecin::class,
                'choice_label' => function(Medecin $medecin) {
                    return 'Dr. ' . $medecin->getNom() . ' - ' .
                    $medecin->getSpecialite();
                },
                'placeholder' => '-- Sélectionner un médecin --',
                'label' => 'Médecin *',
                'attr' => ['class' => 'form-control'],
            ])
            ->add('rendezVous', EntityType::class, [
                'class' => RendezVous::class,

```

```
        'required' => false,
        'placeholder' => '-- Optionnel --',
        'label' => 'Rendez-vous associé',
        'attr' => ['class' => 'form-control'],
    ])
->add('dateDebut', DateTimeType::class, [
    'widget' => 'single_text',
    'label' => 'Date et heure de début *',
    'attr' => ['class' => 'form-control'],
])
->add('type', ChoiceType::class, [
    'choices' => [
        'Consultation Vidéo' => 'VIDEO',
        'Consultation au Cabinet' => 'CABINET',
        'Consultation Audio' => 'AUDIO',
    ],
    'label' => 'Type de consultation *',
    'attr' => ['class' => 'form-control'],
])
->add('motif', TextareaType::class, [
    'label' => 'Motif de la consultation',
    'required' => false,
    'attr' => ['class' => 'form-control', 'rows' => 3],
])
->add('symptomes', TextareaType::class, [
    'label' => 'Symptômes',
    'required' => false,
    'attr' => ['class' => 'form-control', 'rows' => 3],
])
->add('diagnostic', TextareaType::class, [
    'label' => 'Diagnostic',
    'required' => false,
    'attr' => ['class' => 'form-control', 'rows' => 4],
])
->add('notesConsultation', TextareaType::class, [
    'label' => 'Notes',
    'required' => false,
    'attr' => ['class' => 'form-control', 'rows' => 4],
])
->add('montant', NumberType::class, [
    'label' => 'Montant (TND)',
    'required' => false,
```



```

        {{ form_row(form.dateDebut) }}
    </div>
    <div class="col-md-6 mb-3">
        {{ form_row(form.type) }}
    </div>
</div>

    {{ form_row(form.motif) }}
    {{ form_row(form.symptomes) }}
    {{ form_row(form.diagnostic) }}
    {{ form_row(form.notesConsultation) }}
    {{ form_row(form.montant) }}

    {{ form_rest(form) }}

    <div class="d-flex justify-content-between">
        <a href="{% path('app_consultation_show', {id:
consultation.id}) %}" class="btn btn-secondary">
            <i class="fas fa-times"></i> Annuler
        </a>
        <button class="btn btn-primary" type="submit">
            <i class="fas fa-save"></i> Enregistrer les
modifications
        </button>
    </div>

    {{ form_end(form) }}
</div>
</div>
</div>
{% endblock %}
{% templates/consultation/index.html.twig %}
{% extends 'base.html.twig' %}

{% block title %}Liste des Consultations{% endblock %}

{% block body %}
<div class="card">
    <div class="card-header">
        <h3 class="mb-0">
            <i class="fas fa-calendar-alt"></i> Mes Consultations
        </h3>

```

```

</div>
<div class="card-body">
    {# FONCTIONNALITÉ AVANCÉE : RECHERCHE #}
    <form method="get" class="mb-4">
        <div class="row">
            <div class="col-md-8">
                <div class="input-group">
                    <span class="input-group-text">
                        <i class="fas fa-search"></i>
                    </span>
                    <input type="text" name="search"
class="form-control"
                        placeholder="Rechercher par patient,
médecin, diagnostic..."
                        value="{{ search }}">
                    <button class="btn btn-primary" type="submit">
                        Rechercher
                    </button>
                </div>
            </div>
            <div class="col-md-4 text-end">
                <a href="{{ path('app_consultation_new') }}"
class="btn btn-success">
                    <i class="fas fa-plus"></i> Nouvelle
Consultation
                </a>
            </div>
        </div>
    </form>

    {# FONCTIONNALITÉ AVANCÉE : TRI #}
    <div class="mb-3">
        <strong>Trier par :</strong>
        <a href="{{ path('app_consultation_index', {sort:
'dateDebut', order: order == 'ASC' ? 'DESC' : 'ASC'}) }}"
class="btn btn-sm btn-outline-primary">
            Date {{ sort == 'dateDebut' ? (order == 'ASC' ? '↑' :
'↓') : '' }}
        </a>
        <a href="{{ path('app_consultation_index', {sort: 'statut',
order: order == 'ASC' ? 'DESC' : 'ASC'}) }}"
class="btn btn-sm btn-outline-primary">

```

```

Statut {{ sort == 'statut' ? (order == 'ASC' ? '↑' :
'↓') : '' }}
</a>
<a href="{{ path('app_consultation_index', {sort: 'type',
order: order == 'ASC' ? 'DESC' : 'ASC'}) }}"
class="btn btn-sm btn-outline-primary">
Type {{ sort == 'type' ? (order == 'ASC' ? '↑' : '↓') :
'' }}
</a>
</div>

{% if consultations|length > 0 %}
<div class="table-responsive">
<table class="table table-hover">
<thead class="table-light">
<tr>
<th>#</th>
<th>Patient</th>
<th>Médecin</th>
<th>Date</th>
<th>Type</th>
<th>Statut</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
{% for consultation in consultations %}
<tr>
<td>{{ consultation.id }}</td>
<td>
<i class="fas fa-user-injured
text-primary"></i>
{{ consultation.patient }}
</td>
<td>
<i class="fas fa-user-md
text-success"></i>
{{ consultation.medecin }}
</td>
<td>{{ consultation.dateDebut|date('d/m/Y
H:i') }}</td>
<td>
{% if consultation.type == 'VIDEO' %}

```

```

                                <span class="badge bg-primary">
                                    <i class="fas fa-video"></i>
Vidéo

                                </span>
                                {% elseif consultation.type ==
'CABINET' %}

                                <span class="badge bg-info">
                                    <i class="fas fa-hospital"></i>
Cabinet

                                </span>
                                {% else %}

                                <span class="badge bg-secondary">
                                    <i class="fas fa-phone"></i>
Audio

                                </span>
                                {% endif %}
                                </td>
                                <td>
                                    {% set statusClass = {
                                        'PLANIFIEE': 'warning',
                                        'EN_COURS': 'info',
                                        'TERMINEE': 'success',
                                        'ANNULEE': 'danger',
                                        'NO_SHOW': 'dark'
                                    } %}
                                    <span class="badge bg-{{
statusClass[consultation.statut] ?? 'secondary' }}">
                                        {{ consultation.statut }}
                                    </span>
                                </td>
                                <td>
                                    <a href="{{
path('app_consultation_show', {id: consultation.id}) }}"
                                        class="btn btn-sm btn-info"
title="Voir">
                                        <i class="fas fa-eye"></i>
                                    </a>
                                    <a href="{{
path('app_consultation_edit', {id: consultation.id}) }}"
                                        class="btn btn-sm btn-warning"
title="Modifier">
                                        <i class="fas fa-edit"></i>
                                    </a>

```



```

                </td>
            </tr>
        {% endfor %}
    </tbody>
</table>
</div>
{% else %}
    <div class="alert alert-info text-center">
        <i class="fas fa-info-circle"></i>
        Aucune consultation trouvée.
    </div>
{% endif %}
</div>
{% endblock %}

```

```

<?php
// src/Form/LigneOrdonnanceType.php

namespace App\Form;

use App\Consultation_Online_Module_D_Entity\Entity\LigneOrdonnance;
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\Extension\Core\Type\ChoiceType;
use Symfony\Component\Form\Extension\Core\Type\IntegerType;
use Symfony\Component\Form\Extension\Core\Type\TextType;
use Symfony\Component\Form\Extension\Core\Type\TextareaType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;

class LigneOrdonnanceType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array
$options): void
    {
        $builder
            ->add('nomMedicament', TextType::class, [
                'label' => 'Médicament *',
                'attr' => ['class' => 'form-control'],
            ])
            ->add('dosage', TextType::class, [
                'label' => 'Dosage *',

```

```

        'attr' => ['class' => 'form-control', 'placeholder' =>
'Ex: 500mg'],
    ])
    ->add('forme', ChoiceType::class, [
        'choices' => [
            'Comprimé' => 'COMPRIME',
            'Gélule' => 'GELULE',
            'Sirop' => 'SIROP',
            'Ampoule' => 'AMPOULE',
            'Pommade' => 'POMMADE',
            'Crème' => 'CREME',
            'Gouttes' => 'GOUTTES',
            'Suppositoire' => 'SUPPOSITOIRE',
        ],
        'label' => 'Forme *',
        'attr' => ['class' => 'form-control'],
    ])
    ->add('quantite', IntegerType::class, [
        'label' => 'Quantité *',
        'attr' => ['class' => 'form-control', 'min' => 1],
    ])
    ->add('frequence', TextType::class, [
        'label' => 'Fréquence *',
        'attr' => ['class' => 'form-control', 'placeholder' =>
'Ex: 3 fois par jour'],
    ])
    ->add('dureeTraitement', IntegerType::class, [
        'label' => 'Durée (jours) *',
        'attr' => ['class' => 'form-control', 'min' => 1],
    ])
    ->add('instructions', TextareaType::class, [
        'label' => 'Instructions',
        'required' => false,
        'attr' => ['class' => 'form-control', 'rows' => 2],
    ])
    ;
}

public function configureOptions(OptionsResolver $resolver): void
{
    $resolver->setDefaults([
        'data_class' => LigneOrdonnance::class,
    ]);
}

```

```
}  
}
```

```
{# templates/consultation/new.html.twig #}  
{% extends 'base.html.twig' %}  
  
{% block title %}Nouvelle Consultation{% endblock %}  
  
{% block body %}  
<div class="row justify-content-center">  
  <div class="col-md-8">  
    <div class="card">  
      <div class="card-header">  
        <h3 class="mb-0">  
          <i class="fas fa-plus-circle"></i> Nouvelle  
Consultation  
        </h3>  
      </div>  
      <div class="card-body">  
        {{ form_start(form) }}  
  
        <div class="row">  
          <div class="col-md-6 mb-3">  
            {{ form_label(form.patient) }}  
            {{ form_widget(form.patient) }}  
            {{ form_errors(form.patient) }}  
          </div>  
          <div class="col-md-6 mb-3">  
            {{ form_label(form.medecin) }}  
            {{ form_widget(form.medecin) }}  
            {{ form_errors(form.medecin) }}  
          </div>  
        </div>  
  
        <div class="row">  
          <div class="col-md-6 mb-3">  
            {{ form_label(form.dateDebut) }}  
            {{ form_widget(form.dateDebut) }}  
            {{ form_errors(form.dateDebut) }}  
          </div>  
          <div class="col-md-6 mb-3">  
            {{ form_label(form.type) }}  
            {{ form_widget(form.type) }}
```

```

        {{ form_errors(form.type) }}
    </div>
</div>

<div class="mb-3">
    {{ form_label(form.rendezVous) }}
    {{ form_widget(form.rendezVous) }}
    {{ form_errors(form.rendezVous) }}
    <small class="form-text text-muted">Optionnel :
    lier à un rendez-vous existant</small>
</div>

<div class="mb-3">
    {{ form_label(form.motif) }}
    {{ form_widget(form.motif) }}
    {{ form_errors(form.motif) }}
</div>

<div class="mb-3">
    {{ form_label(form.symptomes) }}
    {{ form_widget(form.symptomes) }}
    {{ form_errors(form.symptomes) }}
</div>

<div class="mb-3">
    {{ form_label(form.diagnostic) }}
    {{ form_widget(form.diagnostic) }}
    {{ form_errors(form.diagnostic) }}
</div>

<div class="mb-3">
    {{ form_label(form.notesConsultation) }}
    {{ form_widget(form.notesConsultation) }}
    {{ form_errors(form.notesConsultation) }}
</div>

<div class="mb-3">
    {{ form_label(form.montant) }}
    {{ form_widget(form.montant) }}
    {{ form_errors(form.montant) }}
</div>

    {{ form_rest(form) }}

```

```

        <div class="d-flex justify-content-between">
            <a href="{{ path('app_consultation_index') }}"
class="btn btn-secondary">
                <i class="fas fa-times"></i> Annuler
            </a>
            <button class="btn btn-primary" type="submit">
                <i class="fas fa-save"></i> Créer la
consultation
            </button>
        </div>

        {{ form_end(form) }}
    </div>
</div>
</div>
{% endblock %}

```

```

<?php
// src/Form/OrdonnanceType.php

namespace App\Consultation_Online_Module_D_Forums\Form;

use App\Consultation_Online_Module_D_Entity\Entity\Ordonnance;
use App\Consultation_Online_Module_D_Entity\Entity\LigneOrdonnance;
use App\Consultation_Online_Module_D_Entity\Entity\Patient;
use App\Consultation_Online_Module_D_Entity\Entity\Medecin;
use App\Consultation_Online_Module_D_Entity\Entity\Consultation;
use Symfony\Bridge\Doctrine\Form\Type\EntityType;
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\Extension\Core\Type\CollectionType;
use App\Form\LigneOrdonnanceType;
use Symfony\Component\Form\Extension\Core\Type\DateType;
use Symfony\Component\Form\Extension\Core\Type\TextareaType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;

class OrdonnanceType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array
$options): void
    {

```

```

$builder
    ->add('consultation', EntityType::class, [
        'class' => Consultation::class,
        'choice_label' => function(Consultation $c) {
            return 'Consultation du ' .
$c->getDateDebut()->format('d/m/Y');
        },
        'label' => 'Consultation *',
        'attr' => ['class' => 'form-control'],
    ])
    ->add('patient', EntityType::class, [
        'class' => Patient::class,
        'choice_label' => function(Patient $p) {
            return $p->getPrenom() . ' ' . $p->getNom();
        },
        'label' => 'Patient *',
        'attr' => ['class' => 'form-control'],
    ])
    ->add('medecin', EntityType::class, [
        'class' => Medecin::class,
        'choice_label' => function(Medecin $m) {
            return 'Dr. ' . $m->getNom();
        },
        'label' => 'Médecin *',
        'attr' => ['class' => 'form-control'],
    ])
    ->add('dateEmission', DateType::class, [
        'widget' => 'single_text',
        'label' => 'Date d\'émission *',
        'attr' => ['class' => 'form-control'],
    ])
    ->add('dateValidite', DateType::class, [
        'widget' => 'single_text',
        'label' => 'Date de validité *',
        'attr' => ['class' => 'form-control'],
    ])
    ->add('diagnostic', TextareaType::class, [
        'label' => 'Diagnostic',
        'required' => false,
        'attr' => ['class' => 'form-control', 'rows' => 3],
    ])
    ->add('instructions', TextareaType::class, [
        'label' => 'Instructions',

```

```

        'required' => false,
        'attr' => ['class' => 'form-control', 'rows' => 3],
    ])
    ->add('lignes', CollectionType::class, [
        'entry_type' => LigneOrdonnanceType::class,
        'allow_add' => true,
        'allow_delete' => true,
        'by_reference' => false,
        'label' => 'Médicaments',
    ])
    ;
}

public function configureOptions(OptionsResolver $resolver): void
{
    $resolver->setDefaults([
        'data_class' => Ordonnance::class,
    ]);
}
}

{# templates/consultation/show.html.twig #}
{% extends 'base.html.twig' %}

{% block title %}Consultation #{{ consultation.id }}{% endblock %}

{% block body %}
<div class="row">
    <div class="col-md-8">
        <div class="card">
            <div class="card-header">
                <h3 class="mb-0">
                    <i class="fas fa-clipboard-list"></i>
                    Détails de la Consultation #{{ consultation.id }}
                </h3>
            </div>
            <div class="card-body">
                <div class="row mb-3">
                    <div class="col-md-6">
                        <strong><i class="fas fa-user-injured"></i>
Patient :</strong><br>
                        {{ consultation.patient }}
                    </div>
                    <div class="col-md-6">

```

```

                <strong><i class="fas fa-user-md"></i> Médecin
:</strong><br>
                {{ consultation.medecin }}
            </div>
        </div>

        <div class="row mb-3">
            <div class="col-md-6">
                <strong><i class="fas fa-calendar"></i> Date
:</strong><br>
                {{ consultation.dateDebut|date('d/m/Y à H:i')
}}
            </div>
            <div class="col-md-6">
                <strong><i class="fas fa-hourglass-half"></i>
Durée :</strong><br>
                {% if consultation.dureeMinutes %}
                    {{ consultation.dureeMinutes }} minutes
                {% else %}
                    <em class="text-muted">Non définie</em>
                {% endif %}
            </div>
        </div>

        <div class="row mb-3">
            <div class="col-md-6">
                <strong><i class="fas fa-video"></i> Type
:</strong><br>
                <span class="badge bg-primary">{{
consultation.type }}</span>
            </div>
            <div class="col-md-6">
                <strong><i class="fas fa-flag"></i> Statut
:</strong><br>
                <span class="badge bg-success">{{
consultation.statut }}</span>
            </div>
        </div>

        {% if consultation.motif %}
            <div class="mb-3">
                <strong><i class="fas fa-comment-medical"></i>
Motif :</strong>

```



```

        <p class="mt-2">{{ consultation.motif }}</p>
    </div>
    {% endif %}

    {% if consultation.symptomes %}
    <div class="mb-3">
        <strong><i class="fas fa-thermometer"></i>
Symptômes :</strong>
        <p class="mt-2">{{ consultation.symptomes }}</p>
    </div>
    {% endif %}

    {% if consultation.diagnostic %}
    <div class="mb-3">
        <strong><i class="fas fa-notes-medical"></i>
Diagnostic :</strong>
        <p class="mt-2 border p-3 bg-light rounded">{{
consultation.diagnostic }}</p>
    </div>
    {% endif %}

    {% if consultation.notesConsultation %}
    <div class="mb-3">
        <strong><i class="fas fa-sticky-note"></i> Notes
: </strong>
        <p class="mt-2">{{ consultation.notesConsultation
}}</p>
    </div>
    {% endif %}

    {% if consultation.montant %}
    <div class="mb-3">
        <strong><i class="fas fa-money-bill-wave"></i>
Montant :</strong>
        <span class="fs-4 text-success">{{
consultation.montant }} TND</span>
        {% if consultation.payee %}
            <span class="badge bg-success
ms-2">Payée</span>
        {% else %}
            <span class="badge bg-warning ms-2">Non
payée</span>
        {% endif %}
    </div>
    {% endif %}

```

```

        </div>
        {% endif %}
    </div>
</div>

{# ORDONNANCE ASSOCIÉE #}
{% if consultation.ordonnance %}
<div class="card mt-3">
    <div class="card-header bg-success text-white">
        <h5 class="mb-0">
            <i class="fas fa-prescription"></i> Ordonnance
        </h5>
    </div>
    <div class="card-body">
        <p><strong>Numéro :</strong> {{
consultation.ordonnance.numeroOrdonnance }}</p>
        <p><strong>Date d'émission :</strong> {{
consultation.ordonnance.dateEmission|date('d/m/Y') }}</p>
        <p><strong>Validité :</strong> {{
consultation.ordonnance.dateValidite|date('d/m/Y') }}</p>

        <a href="{{ path('app_ordonnance_show', {id:
consultation.ordonnance.id}) }}"
            class="btn btn-success">
            <i class="fas fa-eye"></i> Voir l'ordonnance
        </a>
    </div>
</div>
{% endif %}
</div>

{# ACTIONS SIDEBAR #}
<div class="col-md-4">
    <div class="card">
        <div class="card-header">
            <h5 class="mb-0"><i class="fas fa-cogs"></i>
Actions</h5>
        </div>
        <div class="card-body">
            <a href="{{ path('app_consultation_edit', {id:
consultation.id}) }}"
                class="btn btn-warning w-100 mb-2">
                <i class="fas fa-edit"></i> Modifier

```

```

        </a>

        {% if consultation.statut == 'PLANIFIEE' %}
        <form method="post" action="{{
path('app_consultation_demarrer', {id: consultation.id}) }}"
        class="d-inline w-100">
            <input type="hidden" name="_token" value="{{
csrf_token('demarrer' ~ consultation.id) }}">
            <button class="btn btn-primary w-100 mb-2">
                <i class="fas fa-play"></i> Démarrer
            </button>
        </form>
        {% endif %}

        {% if consultation.statut == 'EN_COURS' %}
        <form method="post" action="{{
path('app_consultation_terminer', {id: consultation.id}) }}"
        class="d-inline w-100">
            <input type="hidden" name="_token" value="{{
csrf_token('terminer' ~ consultation.id) }}">
            <button class="btn btn-success w-100 mb-2">
                <i class="fas fa-check"></i> Terminer
            </button>
        </form>
        {% endif %}

        {% if not consultation.ordonnance and
consultation.statut == 'TERMINEE' %}
        <a href="{{ path('app_ordonnance_new') }}" class="btn
btn-info w-100 mb-2">
            <i class="fas fa-prescription"></i> Créer
Ordonnance
        </a>
        {% endif %}

        <form method="post" action="{{
path('app_consultation_delete', {id: consultation.id}) }}"
        onsubmit="return confirm('Êtes-vous sûr de
vouloir supprimer cette consultation ?');"
        class="w-100">
            <input type="hidden" name="_token" value="{{
csrf_token('delete' ~ consultation.id) }}">
            <button class="btn btn-danger w-100">

```

```

        <i class="fas fa-trash"></i> Supprimer
    </button>
</form>

<hr>

    <a href="{{ path('app_consultation_index') }}"
class="btn btn-secondary w-100">
        <i class="fas fa-arrow-left"></i> Retour à la liste
    </a>
</div>
</div>
</div>
{% endblock %}

```

```

{# templates/base.html.twig #}
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>{% block title %}MEDISMART - Consultation en ligne{%
endblock %}</title>

    {# Bootstrap CSS #}
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.m
in.css" rel="stylesheet">

    {# Font Awesome #}
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all
.min.css">

    {# CSS personnalisé #}
    <style>
        body {
            font-family: 'Segoe UI', Tahoma, Geneva, Verdana,
sans-serif;
            background-color: #f8f9fa;

```

```
}  
.navbar {  
  background: linear-gradient(135deg, #667eea 0%, #764ba2  
100%);  
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);  
}  
.navbar-brand {  
  font-weight: bold;  
  font-size: 1.5rem;  
}  
.nav-link {  
  color: white !important;  
  margin: 0 10px;  
  transition: all 0.3s;  
}  
.nav-link:hover {  
  color: #ffd700 !important;  
  transform: translateY(-2px);  
}  
.container-main {  
  margin-top: 30px;  
  margin-bottom: 30px;  
}  
.card {  
  border: none;  
  border-radius: 10px;  
  box-shadow: 0 4px 6px rgba(0,0,0,0.1);  
  margin-bottom: 20px;  
}  
.card-header {  
  background: linear-gradient(135deg, #667eea 0%, #764ba2  
100%);  
  color: white;  
  border-radius: 10px 10px 0 0 !important;  
  padding: 15px 20px;  
}  
.btn-primary {  
  background: linear-gradient(135deg, #667eea 0%, #764ba2  
100%);  
  border: none;  
  transition: all 0.3s;  
}  
.btn-primary:hover {
```

```

        transform: translateY(-2px);
        box-shadow: 0 4px 8px rgba(0,0,0,0.2);
    }
    .badge {
        padding: 8px 12px;
        border-radius: 20px;
    }
    .alert {
        border-radius: 10px;
        border: none;
    }
    footer {
        background-color: #343a40;
        color: white;
        padding: 20px 0;
        margin-top: 50px;
    }
</style>

{% block stylesheets %}{% endblock %}
</head>
<body>
    {# NAVBAR - FRONT OFFICE #}
    <nav class="navbar navbar-expand-lg navbar-dark">
        <div class="container">
            <a class="navbar-brand" href="{{
path('app_consultation_index') }}">
                <i class="fas fa-heartbeat"></i> MEDISMART
            </a>
            <button class="navbar-toggler" type="button"
data-bs-toggle="collapse" data-bs-target="#navbarNav">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ms-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="{{
path('app_consultation_index') }}">
                            <i class="fas fa-calendar-check"></i>
Consultations
                        </a>
                    </li>
                    <li class="nav-item">

```

```

        <a class="nav-link" href="{{
path('app_ordonnance_index') }}">
            <i class="fas fa-prescription"></i>
Ordonnances
        </a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="{{
path('app_consultation_new') }}">
            <i class="fas fa-plus-circle"></i> Nouvelle
Consultation
        </a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="#">
            <i class="fas fa-user-circle"></i> Profil
        </a>
    </li>
</ul>
</div>
</div>
</nav>

{# MESSAGES FLASH #}
<div class="container mt-3">
    {% for label, messages in app.flashes %}
        {% for message in messages %}
            <div class="alert alert-{{ label == 'error' ? 'danger'
: label }} alert-dismissible fade show" role="alert">
                <i class="fas fa-{{ label == 'success' ?
'check-circle' : 'exclamation-triangle' }}"></i>
                {{ message }}
                <button type="button" class="btn-close"
data-bs-dismiss="alert"></button>
            </div>
        {% endfor %}
    {% endfor %}
</div>

{# CONTENU PRINCIPAL #}
<div class="container container-main">
    {% block body %}{% endblock %}
</div>

```

```

    {# FOOTER #}
    <footer class="text-center">
        <div class="container">
            <p class="mb-0">
                <i class="fas fa-heart text-danger"></i>
                MEDISMART &copy; 2026 - Module D : Consultation en
ligne
            </p>
            <small>Développé avec Symfony 7</small>
        </div>
    </footer>

    {# Bootstrap JS #}
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

    {% block javascripts %}{% endblock %}
</body>
</html>

```

D:\PI-JAVAFX-SYMFONY-WEB-DESKTOP\symfony\templates\Consultation_Online_Module_D_Forumslordonnance

```

{# templates/ordonnance/index.html.twig #}
{% extends 'base.html.twig' %}

{% block title %}Ordonnances{% endblock %}

{% block body %}
<div class="card">
    <div class="card-header">
        <h3><i class="fas fa-prescription"></i> Mes Ordonnances</h3>
    </div>
    <div class="card-body">
        <a href="{{ path('app_ordonnance_new') }}" class="btn
btn-success mb-3">
            <i class="fas fa-plus"></i> Nouvelle Ordonnance
        </a>
    </div>

```



```

        <table class="table table-hover">
            <thead>
                <tr>
                    <th>Numéro</th>
                    <th>Patient</th>
                    <th>Médecin</th>
                    <th>Date émission</th>
                    <th>Validité</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                {% for ordonnance in ordonnances %}
                <tr>
                    <td>{{ ordonnance.numeroOrdonnance }}</td>
                    <td>{{ ordonnance.patient }}</td>
                    <td>{{ ordonnance.medecin }}</td>
                    <td>{{ ordonnance.dateEmission|date('d/m/Y')
}}</td>

                    <td>{{ ordonnance.dateValidite|date('d/m/Y')
}}</td>

                    <td>
                        <a href="{% path('app_ordonnance_show', {id:
ordonnance.id}) %}" class="btn btn-sm btn-info">
                            <i class="fas fa-eye"></i>
                        </a>
                    </td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</div>
{% endblock %}

```

```

{# templates/ordonnance/index.html.twig #}
{% extends 'base.html.twig' %}

{% block title %}Ordonnances{% endblock %}

{% block body %}
<div class="card">
    <div class="card-header">

```

```

        <h3><i class="fas fa-prescription"></i> Mes Ordonnances</h3>
    </div>
    <div class="card-body">
        <a href="{% path('app_ordonnance_new') %}" class="btn
btn-success mb-3">
            <i class="fas fa-plus"></i> Nouvelle Ordonnance
        </a>

        <table class="table table-hover">
            <thead>
                <tr>
                    <th>Numéro</th>
                    <th>Patient</th>
                    <th>Médecin</th>
                    <th>Date émission</th>
                    <th>Validité</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                {% for ordonnance in ordonnances %}
                <tr>
                    <td>{{ ordonnance.numeroOrdonnance }}</td>
                    <td>{{ ordonnance.patient }}</td>
                    <td>{{ ordonnance.medecin }}</td>
                    <td>{{ ordonnance.dateEmission|date('d/m/Y')
}}</td>
                    <td>{{ ordonnance.dateValidite|date('d/m/Y')
}}</td>
                    <td>
                        <a href="{% path('app_ordonnance_show', {id:
ordonnance.id}) %}" class="btn btn-sm btn-info">
                            <i class="fas fa-eye"></i>
                        </a>
                    </td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</div>
{% endblock %}
{# templates/ordonnance/index.html.twig #}

```

```
{% extends 'base.html.twig' %}

{% block title %}Ordonnances{% endblock %}

{% block body %}
<div class="card">
    <div class="card-header">
        <h3><i class="fas fa-prescription"></i> Mes Ordonnances</h3>
    </div>
    <div class="card-body">
        <a href="{{ path('app_ordonnance_new') }}" class="btn
btn-success mb-3">
            <i class="fas fa-plus"></i> Nouvelle Ordonnance
        </a>

        <table class="table table-hover">
            <thead>
                <tr>
                    <th>Numéro</th>
                    <th>Patient</th>
                    <th>Médecin</th>
                    <th>Date émission</th>
                    <th>Validité</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                {% for ordonnance in ordonnances %}
                <tr>
                    <td>{{ ordonnance.numeroOrdonnance }}</td>
                    <td>{{ ordonnance.patient }}</td>
                    <td>{{ ordonnance.medecin }}</td>
                    <td>{{ ordonnance.dateEmission|date('d/m/Y')
}}</td>
                    <td>{{ ordonnance.dateValidite|date('d/m/Y')
}}</td>
                    <td>
                        <a href="{{ path('app_ordonnance_show', {id:
ordonnance.id}) }}" class="btn btn-sm btn-info">
                            <i class="fas fa-eye"></i>
                        </a>
                    </td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</div>
{% endblock %}
```

```
        {% endfor %}
    </tbody>
</table>
</div>
</div>
{% endblock %}
```