Création des dossiers médicaux : Points clés et processus

- Il semble probable que la création d'un dossier médical implique un docteur authentifié, un patient sélectionné, et la saisie de notes, avec une association possible d'images DICOM via Orthanc.
- Les entités principales sont **User** (docteur et patient) et **MedicalRecord**, avec **MedicalImage** pour les images, mais sans lien direct avec les dossiers.

Authentification et autorisation

Un docteur doit se connecter via Spring Security avec JWT pour accéder à la fonctionnalité. Seuls les utilisateurs avec le rôle DOCTEUR peuvent créer des dossiers.

Sélection du patient et saisie des données

Le docteur choisit un patient via une interface React, puis entre les notes médicales dans un formulaire, incluant des détails comme le diagnostic ou le traitement.

Enregistrement du dossier

Le système envoie une requête au backend, qui crée une entité **MedicalRecord**, lie le patient et le docteur, fixe la date actuelle, enregistre les notes, et sauvegarde le tout dans PostgreSQL.

Association d'images (optionnel)

Les images DICOM peuvent être téléchargées séparément vers Orthanc, avec leurs identifiants stockés dans **MedicalImage**, liés au patient, mais sans lien direct au dossier médical.

Rapport détaillé sur la création des dossiers médicaux

Ce rapport explore en détail le processus de création des dossiers médicaux dans le cadre de l'application médicale développée avec Spring Boot, React.js, et l'intégration d'Orthanc pour les images DICOM. Il couvre les entités impliquées, les étapes du processus, et les considérations techniques, en s'appuyant sur la conception décrite.

Contexte et objectifs

L'application vise à gérer des dossiers médicaux, avec des entités principales comme **User**, **MedicalRecord**, et **MedicalImage**, intégrant Orthanc pour le stockage des fichiers DICOM. La

création d'un dossier médical est une fonctionnalité clé, accessible uniquement aux utilisateurs avec le rôle DOCTEUR, et doit être robuste et évolutive.

Entités impliquées

Les entités directement impliquées dans la création d'un dossier médical sont :

- **User** : Représente les utilisateurs, avec des rôles comme DOCTEUR et PATIENT. Le docteur est l'utilisateur authentifié créant le dossier, et le patient est le sujet du dossier. Le champ `dicomPatientId` peut être utilisé pour lier les fichiers DICOM.
- **MedicalRecord** : Stocke les informations du dossier, incluant l'ID du patient, l'ID du docteur (optionnel), la date de création, et les notes. C'est l'entité centrale créée lors du processus.
- **MedicalImage** : Bien que liée au patient via des identifiants Orthanc (`instanceId`), elle n'est pas directement liée au **MedicalRecord**, mais peut être associée indirectement via les notes ou consultée en relation avec le patient.

Processus détaillé de création

1. **Authentification et autorisation** :
   - Le docteur doit se connecter via une interface React, utilisant Spring Security avec JWT pour l'authentification. L'annotation `@PreAuthorize("hasRole('DOCTOR')")` dans le contrôleur garantit que seul un docteur peut créer des dossiers.
   - Exemple d'endpoint : `/api/medical-records` , protégé par JWT.
2. **Sélection du patient et saisie des données** :
   - Via une interface React, comme le composant `MedicalRecordForm` , le docteur sélectionne un patient (via son ID) et entre les notes dans un formulaire. Par exemple :

```
function MedicalRecordForm({ patientId }) {
    const [notes, setNotes] = useState('');
    const handleSubmit = async () => {
        await axios.post('/api/medical-records', { patientId, notes });
    };
    return (
        <div>
            <textarea value={notes} onChange={(e) =>
setNotes(e.target.value)} />
            <button onClick={handleSubmit}>Enregistrer</button>
        </div>
    );
}
```

- Les notes peuvent inclure des détails comme le diagnostic, le traitement, ou des références à des images DICOM.

3. **Envoi de la requête et traitement backend** :
   - Le frontend envoie une requête POST à `/api/medical-records` avec l'ID du patient et les notes.
   - Le backend, via le contrôleur `MedicalRecordController`, appelle le service `MedicalRecordService` pour créer le dossier :

```
@RestController
@RequestMapping("/api/medical-records")
public class MedicalRecordController {
    @Autowired
    private MedicalRecordService service;
    @PostMapping
    @PreAuthorize("hasRole('DOCTOR')")
    public MedicalRecordDTO create(@RequestBody MedicalRecordDTO dto) {
        return service.createMedicalRecord(dto);
    }
}
```

   - Le service récupère le patient, crée une nouvelle entité **MedicalRecord**, et la sauvegarde :

```
@Service
public class MedicalRecordService {
    @Autowired
    private MedicalRecordRepository repository;
    @Autowired
    private UserRepository userRepository;
    public MedicalRecordDTO createMedicalRecord(MedicalRecordDTO dto) {
        User patient = userRepository.findById(dto.getPatientId())
                .orElseThrow(() -> new RuntimeException("Patient not
found"));
        MedicalRecord record = new MedicalRecord();
        record.setPatient(patient);
        record.setCreationDate(LocalDateTime.now());
        record.setNotes(dto.getNotes());
        repository.save(record);
        return convertToDTO(record);
    }
}
```

- Note : Le docteur est implicitement défini comme l'utilisateur authentifié, bien que dans le code fourni, il n'est pas explicitement défini. Il est probable que cela soit géré via Spring Security, par exemple en passant l'utilisateur authentifié dans la méthode.

4. **Sauvegarde dans la base de données** :
   - Le dossier est sauvegardé dans PostgreSQL, avec les champs `patient`, `doctor` (optionnel), `creationDate`, et `notes`. Les champs comme `instanceId` pour les images DICOM ne sont pas directement stockés dans **MedicalRecord**, mais dans **MedicalImage**.
5. **Association d'images DICOM (optionnel)** :
   - La création du dossier médical ne nécessite pas directement l'upload d'images, mais celles-ci peuvent être associées au patient via un processus séparé. Par exemple, via le composant `DicomUpload` :

```
function DicomUpload({ patientId }) {
    const [file, setFile] = useState(null);
    const handleFileChange = (event) => {
        setFile(event.target.files[0]);
    };
    const handleUpload = async () => {
        const formData = new FormData();
        formData.append('file', file);
        await axios.post(`/api/dicom/upload/${patientId}`, formData, {
            headers: { 'Content-Type': 'multipart/form-data' }
        });
        alert('Fichier DICOM uploadé avec succès');
    };
    return (
        <div className="p-4">
            <input type="file" onChange={handleFileChange}
classNAme="mb-2" />
            <button onClick={handleUpload} className="bg-blue-500 text-
white p-2 rounded">
                Uploader
            </button>
        </div>
    );
}
```

   - Les fichiers sont envoyés à Orthanc via `/instances`, et les `instanceId` sont stockés dans **MedicalImage**, liés au patient. Les annotations peuvent être ajoutées via `/api/medical-images/{patientId}/annotate`.

Considérations techniques

- **Liens entre entités** : Il n'y a pas de relation directe entre **MedicalRecord** et **MedicalImage** dans la base de données. Les images sont associées au patient, et le dossier médical peut référencer des images via les notes ou être consulté avec les images du patient.
- **Sécurité** : L'accès est restreint aux docteurs via `@PreAuthorize`, et les données sensibles sont protégées par JWT.
- **Évolutivité** : Le système permet d'ajouter des annotations aux images et de gérer les dossiers séparément, facilitant l'ajout de fonctionnalités futures.

Tableau récapitulatif des entités et rôles

Entité	Rôle dans la création du dossier médical	Champs principaux
User (DOCTEUR)	Créateur du dossier, authentifié via JWT	id, username, role, dicomPatientId
User (PATIENT)	Sujet du dossier, sélectionné par le docteur	id, username, role, dicomPatientId
MedicalRecord	Entité créée, stocke les informations du dossier	id, patient, doctor, creationDate, notes
MedicalImage	Non directement lié, gère les images DICOM du patient	id, patient, instanceId, annotations

Conclusion

Le processus de création des dossiers médicaux implique principalement les entités **User** (docteur et patient) et **MedicalRecord**, avec un workflow clair d'authentification, de saisie, et d'enregistrement. Les images DICOM, gérées via **MedicalImage** et Orthanc, sont associées au patient séparément, sans lien direct au dossier, mais peuvent être référencées via les notes. Cette structure assure une gestion robuste et évolutive, conforme aux besoins décrits.

Key Citations

- Spring Boot Official Documentation
- React Official Documentation
- Orthanc Server User Manual
- PostgreSQL Official Documentation