```
docker-compose down
docker-compose up -d --build
docker volume prune
docker logs -f byteco-backend-gestionsinistres-1
docker exec -it byteco-backend-gestionsinistres-1 sh

 ls -l ./uploads/constats/
docker exec -it byteco-backend-gestionsinistres-1 bash

curl -X POST http://localhost:8085/api/sinistres/accident   -H "Authorization: Bearer eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJ1c2VyNEBleGFtcGxlLmNvbSIsInJvbGUiOiJVU0VSIiwiaWF0IjoxNzQzMTY4MTYwLCJleHAiOjE3NDMyNTQ1NjB9.dzyr1lIUT4s4lH_IA-KjHuuVh7EH485Nx5uXsL8yKlkKafTwDKWnZuUTIQYWk9ZluTQK9SRCG_8j0QjOOubb5A"   -H "Content-Type: multipart/form-data"   -F 'dto={"description":"Accident test final", "dateDeclaration":"2025-03-28T17:00:00"};type=application/json'   -F "constat=@/home/m2ipint/patricia.totoummandoum.etu/Downloads/TP.pdf"

curl -X GET http://localhost:8085/api/sinistres/accident/uploads/constats/95fbf9e0-d5be-4c3c-9f6e-57ab697d03cc-TP.pdf --output mon_constat_final.pdf
```

- 🔐 Ou un `SinistreAdminController` (admin peut voir tous les sinistres, les filtrer, changer statut, etc.) ?

Passons, démarrons la déclaaration d'incident accessible seulement lorsqu'on est connecté. L'onglet et la page Assistance et Sinistre est accessible de tous, mais lorsque je clique sur une carte, par exemple Déclarer un sinistre, si on est pas connectée, ca nous renvoie vers le formulaire de connexion, si la connexion fonctionne, on peut voir la page de déclaration d'incident, que nous ajoutons plutot dans le dashboard du client, uniquement.

## 🎯 D'où vient `utilisateurId` ?

Il vient **du token JWT** émis par ton **microservice d'authentification**.
 Quand un client est connecté, chaque requête envoyée par le frontend contient le token JWT (dans les headers).

---

## 🧩 À quoi ressemble le token JWT ?

Un token typique (dans l'en-tête `Authorization: Bearer <token>`) contient un `payload` comme :

json

CopierModifier

```
{
  "sub": "client@gmail.com",
  "userId": 5,
  "role": "CLIENT",
  "exp": 1712172367
}
```

👉 **`userId`** est ce qu'on récupère du token pour **l'associer à une déclaration de sinistre**.

---

## ✅ Où récupérer `userId` dans Spring Boot ?

Il faut :

1. Lire le token depuis le header
2. Extraire l'`userId` (via un `JwtUtils` ou depuis le `SecurityContext`)
3. L'injecter dans la méthode de création de sinistre

---

## 🔧 Exemple : Contrôleur `SinistreController`

java

CopierModifier

```
@PostMapping
```

```
public ResponseEntity<SinistreDTO> creerSinistre(@RequestBody
SinistreDTO dto, @RequestHeader("Authorization") String authHeader)
{

    String token = authHeader.replace("Bearer ", "");

    Long userId = jwtUtils.extractUserId(token); // <--- méthode à
implémenter



    dto.setUtilisateurId(userId);

    SinistreDTO saved = sinistreService.creerSinistre(dto);

    return ResponseEntity.status(HttpStatus.CREATED).body(saved);

}
```

---

### 🔐 Et côté `JwtUtils` ?

Tu dois décoder le token (si tu ne le fais pas déjà dans un filtre de sécurité) :

java

CopierModifier

```
public Long extractUserId(String token) {

    Claims claims = Jwts.parser()

        .setSigningKey(secretKey)

        .parseClaimsJws(token)

        .getBody();



    return claims.get("userId", Long.class);

}
```

---

## 📦 Résumé

| Origine | Explication |
| --- | --- |
| **JWT token** | Transporté automatiquement par le frontend |
| **userId dans token** | Injecté par le microservice Auth |
| **Back `gestionsinistres`** | Doit le récupérer du token pour associer le sinistre à un utilisateur |

C'est normal si il n' ya pas getDetails et setDetails dans le dto package com.glop.gestionsinistres.dto;

import java.util.Date;

public class SinistreDTO {

private Long id;

private String description;

private String type;

private Date dateDeclaration;

private double montantEstime;

private String statut;

private Long utilisateurId;

// === Getters & Setters ===

public Long getId() {

return id;

```
}
public void setId(Long id) {
this.id = id;
}
public String getDescription() {
return description;
}
public void setDescription(String description) {
this.description = description;
}
public String getType() {
return type;
}
public void setType(String type) {
this.type = type;
}
public Date getDateDeclaration() {
return dateDeclaration;
}
public void setDateDeclaration(Date dateDeclaration) {
this.dateDeclaration = dateDeclaration;
}
public double getMontantEstime() {
return montantEstime;
}
public void setMontantEstime(double montantEstime) {
```

this.montantEstime = montantEstime;

}

public String getStatut() {

return statut;

}

public void setStatut(String statut) {

this.statut = statut;

}

public Long getUtilisateurId() {

return utilisateurId;

}

public void setUtilisateurId(Long utilisateurId) {

this.utilisateurId = utilisateurId;

}

} alors qu'il est present dans le entity ?