4. Sécurisation avec middleware

6.1 Qu'est-ce qu'un middleware ?

Un middleware est une fonction qui s'exécute **entre** la réception d'une requête et l'envoi de la réponse. C'est comme un filtre ou un checkpoint.

Utilisations courantes des middlewares :

- Authentification
- Validation des données
- Logging (journalisation)
- Gestion des erreurs
- Transformation des données

6.2 Création du middleware d'authentification

Créez un dossier `middleware` et un fichier `checkApiKey.js` :

```
// Middleware pour vérifier la clé API const checkApiKey = (req, res, next) => { // Récupérer la clé API depuis les headers const apiKey = req.headers['x-api-key']; // Clé API attendue (en production, stockez-la dans des variables d'environnement) const validApiKey = 'ma-super-cle-api-2024'; // Vérification if (!apiKey) { return res.status(401).json({ error: 'Non autorisé', message: 'Clé API manquante. Ajoutez le header x-api-key à votre requête' }); } if (apiKey !== validApiKey) { return res.status(403).json({ error: 'Accès refusé', message: 'Clé API invalide' }); } // Si tout est OK, on passe au prochain middleware/route console.log('✅ Clé API valide'); next(); }; module.exports = checkApiKey;
```

Codes de statut HTTP importants :

- **200** : OK (succès)
- **201** : Created (ressource créée)
- **400** : Bad Request (requête mal formée)
- **401** : Unauthorized (non authentifié)
- **403** : Forbidden (non autorisé)

- **404** : Not Found (ressource non trouvée)
- **500** : Internal Server Error (erreur serveur)

6.3 Application du middleware

Dans `index.js`, importez le middleware et appliquez-le à vos routes :

```
const checkApiKey = require('./middleware/checkApiKey'); // ... autres
imports et config ... // Routes CRUD (protégées par le middleware)
app.get('/api/cars', checkApiKey, carsController.getAllCars);
app.get('/api/cars/:id', checkApiKey, carsController.getCarById);
app.post('/api/cars', checkApiKey, carsController.createCar);
app.put('/api/cars/:id', checkApiKey, carsController.updateCar);
app.delete('/api/cars/:id', checkApiKey, carsController.deleteCar);
```

✏️ **Exercice 9** :

- Créez le middleware `checkApiKey.js`
- Appliquez-le à vos routes
- Testez avec Postman :
  - Sans header → Erreur 401
  - Avec mauvaise clé → Erreur 403
  - Avec bonne clé → Succès

Pour ajouter le header dans Postman :

1. Ouvrez votre requête
2. Allez dans l'onglet "Headers"
3. Ajoutez :
   - Key: `x-api-key`
   - Value: `ma-super-cle-api-2024`