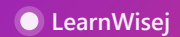


Exercice de code avec l'IA

Construisez l'exemple AccessOps de ce module avec ChatGPT ou Claude à partir d'un prompt prêt à l'emploi qui oriente le modèle vers la spécification du module, puis relisez, exécutez et étendez ce qu'il vous renvoie.



L'assistant, ce sont les mains rapides. L'architecte, c'est toujours vous.

Un assistant peut produire l'exemple de ce module en quelques minutes. Ce qu'il ne peut pas faire, c'est décider si le résultat est la bonne conception pour votre application — cette décision vous revient. C'est vous qui apportez le jugement : briefez-le correctement, puis vérifiez ce qu'il vous renvoie à la lumière de ce que le Module 7 vient de vous apprendre.

Ce que vous demandez

Réalisez le projet de synthèse AccessOps : imposez HTTPS avec HSTS et définissez les en-têtes de sécurité, marquez les cookies Secure et HttpOnly avec une politique SameSite stricte, fixez le délai d'expiration de session et les limites de taille des requêtes, ajoutez une limitation du débit sur la connexion et les liens de partage, lancez un audit des dépendances et consignez les résultats, puis passez toute la checklist de ce cours sur l'application et corrigez chaque constat.

Le module demande ces livrables, et le prompt ci-dessous demande exactement ceux-là à l'assistant :

- ✓ Configuration HTTPS, HSTS et en-têtes de sécurité
- ✓ Paramètres de durcissement des cookies et de la session
- ✓ Limitation du débit sur la connexion et les liens de partage
- ✓ Rapport d'audit des dépendances
- ✓ Checklist de revue de sécurité complétée avec les corrections

Pourquoi il est conçu ainsi

AccessOps vérifie désormais les mots de passe par rapport à un hachage salé, résout les permissions dans un service, n'envoie les documents que par DocumentService et audite chaque approbation. Rien de tout cela n'aide si la première requête circule en clair, si un script peut lire le cookie de session, ou si le build embarque un package faisant l'objet d'un avis de sécurité publié. Le dernier kilomètre revient à l'hébergement: le certificat, les en-têtes de réponse, les attributs des cookies, les limites de requête, et la version de chaque élément du graphe de dépendances.

Voici les termes que l'assistant doit maîtriser. Si ce qu'il vous renvoie les emploie de façon approximative, il n'a pas compris le module :

Terme	Ce qu'il signifie ici
HSTS	Un en-tête Strict-Transport-Security qui demande au navigateur de refuser le HTTP en clair pour cet hôte pendant une durée donnée, afin qu'une visite ultérieure ne soit jamais interceptée lors de la redirection.
Content-Security-Policy	Un en-tête qui liste les origines depuis lesquelles une page peut charger scripts, styles et frames; il transforme un script injecté en requête bloquée au lieu d'un code qui s'exécute dans une session active.
frame-ancestors	La directive CSP (X-Frame-Options en est l'ancien équivalent) qui empêche un autre site d'encadrer AccessOps pour pousser un manager à cliquer sur Approve.
attributs des cookies	Secure tient le cookie de session à l'écart du HTTP en clair, HttpOnly le tient à l'écart de JavaScript, et SameSite l'empêche d'accompagner une requête lancée par un autre site.
limites de requête	Des plafonds, au niveau de l'hébergement, sur la taille du corps et des en-têtes, qui rejettent un upload surdimensionné avant qu'une seule ligne de code d'AccessOps ne s'exécute.
token bucket	Un compteur qui n'accorde une tentative que tant qu'il détient un jeton et qui se recharge à un rythme fixe, de sorte que le trafic peut connaître un léger pic puis ralentit fortement.
audit des dépendances	Une vérification reproductible de chaque package direct et transitif, et du runtime, au regard des avis de sécurité publiés — consignée avec une date et une décision.

Construisez-le — choisissez votre assistant

Les deux boutons ouvrent un nouvel onglet avec le **même prompt prêt à l'emploi** déjà rempli. Le prompt oriente le modèle vers la spécification de ce module — un unique fichier Markdown qui contient la leçon, l'énoncé du lab et les livrables — pour qu'il travaille sur le vrai contenu au lieu de deviner de quoi parle le Module 7. Vous pouvez ouvrir vous-même [module-07.md](#) pour voir exactement ce qu'il lira.



Utiliser ChatGPT

Ouvre [chatgpt.com](#) avec le prompt de ce module déjà chargé. Fonctionne avec GPT-5 ou Codex ; toute offre capable de naviguer sur le web récupérera la spécification.



Utiliser Claude

Ouvre [claude.ai](#) avec le même prompt déjà chargé. Si vous utilisez Claude Code, vous pouvez plutôt le coller dans un terminal ouvert sur un dossier vide.



Envoyez-le tel quel et vous obtenez un projet, pas un extrait

Le prompt se termine en demandant toute la solution sous forme d'un unique `.zip`, avec les étapes pour la compiler et l'exécuter. Téléchargez-le, ouvrez la solution dans Visual Studio, appuyez sur F5 et AccessOps s'ouvre sur `localhost`. Il vous faut le SDK .NET et le package NuGet `Wisej-4`; tout le reste, le prompt demande à l'assistant de l'écrire lui-même.

Je suis le Module 7 - « Checklist de durcissement et projet de synthèse » - du cours LearnWisej « Modèles de sécurité et d'authentification », et je veux construire avec toi l'exemple de ce module. Réponds-moi en français.

Commence par lire la spécification du module. C'est un unique fichier Markdown (en anglais) qui contient la leçon complète, l'énoncé du lab et les livrables :

<https://learnwisej.com/assets/courses/security-and-auth-patterns/specs/module-07.md>

Construis ensuite un projet Wisej.NET 4 complet que je puisse ouvrir dans Visual Studio :

1. Projet « AccessOps », C#, net8.0 ou ultérieur, package NuGet Wisej-4 4.1.x, hôte Kestrel qui appelle `app.UseWisej()`.
2. Construis ce que demande le lab de la spécification : Réalisez le projet de synthèse AccessOps : imposez HTTPS avec HSTS et définissez les en-têtes de sécurité, marquez les cookies Secure et HttpOnly avec une politique SameSite stricte, fixez le délai d'expiration de session et les limites de taille des requêtes, ajoutez une limitation du débit sur la connexion et les liens de partage, lancez un audit des dépendances et consignez les résultats, puis passez toute la checklist de ce cours sur l'application et corrigez chaque constat.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Configuration HTTPS, HSTS et en-têtes de sécurité
 - Paramètres de durcissement des cookies et de la session
 - Limitation du débit sur la connexion et les liens de partage
 - Rapport d'audit des dépendances
 - Checklist de revue de sécurité complétée avec les corrections
4. Suis la spécification partout où elle nomme une classe, une propriété ou un événement Wisej.NET - utilise exactement celui-là. Là où elle laisse le choix, dis-moi quelle API tu as choisie et pourquoi.
5. Garde l'autorité côté serveur : l'état et la validation vivent dans .NET, le navigateur affiche et remonte les informations. Ne déplace jamais une décision dans JavaScript parce que c'est plus simple là-bas.
6. Rends le résultat démontrable : je dois pouvoir l'exécuter et voir le comportement dont traite ce module, y compris ce qui se passe quand quelque chose tourne mal.

Avant d'écrire la moindre ligne de code, reformule avec tes propres mots ce que tu vas construire et où vivra chaque élément, pour que je puisse corriger la conception tant que cela ne coûte pas cher.

Quand le code est terminé, rassemble toute la solution dans un unique fichier .zip téléchargeable et donne-moi les étapes exactes pour la compiler et l'exécuter.

Enfin, réponds à ces questions de la spécification en t'appuyant sur ta propre implémentation :

- Un middleware peut limiter le débit du handler de lien de partage, mais pas celui du bouton Sign in. Qu'est-ce qui distingue ces deux requêtes dans une application Wisej.NET, et où la limite de connexion doit-elle se trouver à la place?
- Votre Content-Security-Policy est en mode bloquant et la grille s'affiche sans styles. Quelle directive en est probablement la cause, et comment le mode report-only vous l'aurait-il signalé avant que le déploiement ne le fasse?
- L'audit des dépendances signale un package vulnérable que rien dans AccessOps ne référence directement. Comment est-il arrivé dans le build, et quelles sont les deux façons honnêtes de clore ce constat?

Pose-moi des questions sur tout ce qui est ambigu avant de commencer.

Puis relisez ce qui vous revient

Un code qui compile n'est pas encore un code juste. Les questions intéressantes sont des questions de conception : est-ce bien le contrôle Wisej.NET que la tâche exige, suit-il le cycle de vie que le framework vous offre, l'état de référence est-il sur le serveur ou a-t-il été laissé dans le navigateur parce que c'était pratique ? Lisez le résultat en relecteur, pas en client.

Ce module vous donne la checklist. Répondez à chaque point en vous appuyant sur le code que vous venez de recevoir :

- ✓ Un middleware peut limiter le débit du handler de lien de partage, mais pas celui du bouton Sign in. Qu'est-ce qui distingue ces deux requêtes dans une application Wisej.NET, et où la limite de connexion doit-elle se trouver à la place?
- ✓ Votre Content-Security-Policy est en mode bloquant et la grille s'affiche sans styles. Quelle directive en est probablement la cause, et comment le mode report-only vous l'aurait-il signalé avant que le déploiement ne le fasse?
- ✓ L'audit des dépendances signale un package vulnérable que rien dans AccessOps ne référence directement. Comment est-il arrivé dans le build, et quelles sont les deux façons honnêtes de clore ce constat?

Tout ce que vous avez dû corriger mérite d'être noté. Une courte note des conventions et des décisions retenues, collée dans le premier message de votre prochaine session, vaut plus que n'importe quel prompt astucieux.



Pas encore d'exemple de référence pour ce cours — le vôtre est le premier

Le [dépôt des exemples LearnWisej](#) contient des solutions complètes pour la plupart des cours, mais pas encore pour celui-ci. Il n'y a donc rien à quoi vous comparer et aucun raccourci : la spécification et votre propre relecture sont votre seul filet de sécurité. Gardez ce que vous construisez — et si le résultat est réussi, nous aimerions le voir.

Trois prompts pour aller plus loin

Orientez votre assistant vers le projet que vous venez de construire et essayez ceux-ci. Chacun est un exercice de relecture et d'extension, pas une réécriture.

- ✓ **Relisez-le comme une pull request.** *"Fais-moi parcourir ce code AccessOps comme si tu relisais la pull request d'un collègue et que tu ne l'avais pas écrit. Où vit l'état, que se passe-t-il de bout en bout lors d'une action de l'utilisateur, que renverrais-tu pour correction, et quelles parties ne serais-tu pas à l'aise de montrer à un client ? Sois précis et cite les lignes dont tu parles — ne me résume pas la conception."*
- ✓ **Remplacez une pièce sans déplacer les jointures.** *"Choisis la partie de ce projet AccessOps qui remplace le plus manifestement quelque chose de réel — une source de données factice, un stockage en mémoire, un service simplifié — et remplace-la par une implémentation digne de la*

production. Le contrat de part et d'autre ne doit pas changer : mêmes types, mêmes appels, même comportement en cas d'échec. Dis-moi ce que l'implémentation réelle t'a obligé à gérer et que la version provisoire ne gérait pas."

- ✓ **Faites-le résister aux mauvais jours.** *"Examine ce projet AccessOps pour repérer les modes de défaillance qu'il ne gère pas encore — une saisie erronée ou manquante, une dépendance lente ou indisponible, un traitement qui aboutit après que l'utilisateur a changé de page, et deux actions qui se chevauchent. Pour chacun, dis-moi d'abord s'il s'agit d'un risque réel dans Wisej.NET 4 et pourquoi, et seulement ensuite écris la correction. Là où tu estimes qu'aucune gestion n'est nécessaire, dis-le et explique ton raisonnement au lieu d'ajouter du code défensif."*



Emportez-le dans le lab

Le lab pratique qui clôt ce module demande les mêmes livrables que ceux que vous venez de faire produire à l'assistant. Servez-vous de ce que vous avez ici comme forme de votre réponse et comme preuve à l'appui — pas comme la réponse elle-même. Le lab est noté sur votre raisonnement, et cette partie doit venir de vous.