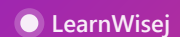


Exercice de code avec l'IA

Construisez l'exemple EnterpriseOps 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 — y compris en le comparant à notre propre solution de référence sur GitHub.



Le même module, construit comme vous construirez vraiment.

Toutes les autres sections de ce module expliquent comment fonctionne « Architecture de sécurité : identité, SSO, autorisation, audit et déploiement sécurisé ». Celle-ci vous demande de le produire — en pilotant un assistant avec la spécification du module elle-même, puis en vérifiant que le résultat s'exécute et fait ce que décrit l'énoncé du lab. C'est la méthode de travail que vous utiliserez sur de vrais projets, mise en pratique sur un sujet dont vous connaissez déjà la bonne réponse.

Ce que vous demandez

Ajoutez une simulation d'authentification d'entreprise, la correspondance des claims vers les permissions, l'application des rôles par tenant, l'autorisation au niveau des services, la revue du HTML sûr, l'approbation des exports et la journalisation d'audit des actions sensibles.

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

- ✓ Conception de la correspondance d'identité
- ✓ Matrice des permissions
- ✓ Implémentation du service de permissions
- ✓ Écran du journal d'audit
- ✓ Checklist de durcissement de la sécurité

Pourquoi il est conçu ainsi

L'architecture côté serveur réduit de nombreuses surfaces d'attaque dans le navigateur, mais les systèmes avancés ont toujours besoin d'intégration de l'identité, d'autorisation au niveau des services, de pistes d'audit, d'un rendu sûr et d'une configuration d'exploitation sécurisée.

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
frontière OIDC/SSO	Le point où un fournisseur d'identité externe remet à l'application une identité vérifiée. Tout ce qui se trouve après cette frontière relève de la responsabilité de l'application.
correspondance des claims	Traduire les claims propres au fournisseur en rôles et permissions de l'application, pour que le code d'autorisation ne dépende jamais du format d'un fournisseur d'identité particulier.
service de permissions	Service unique qui indique si l'utilisateur courant peut effectuer une action, pour que la règle vive en un seul endroit plutôt que dans chaque écran.
autorisation au niveau des services	Appliquer les permissions dans les services applicatifs, et pas seulement dans l'UI, pour qu'un bouton masqué ne soit jamais la seule protection d'une opération.
journal d'audit	Trace infalsifiable de qui a fait quoi et quand pour les actions importantes, afin de pouvoir enquêter sur les événements de sécurité et prouver la conformité.
revue d'AllowHtml	Vérifier chaque endroit où un contrôle autorise du HTML brut pour s'assurer que le contenu fourni par l'utilisateur est assaini, car un HTML non revu constitue un risque de cross-site scripting.
gestion des secrets	Conserver les chaînes de connexion, les clés et les jetons dans une configuration sécurisée ou un coffre-fort, jamais dans le contrôle de code source ni dans des paramètres visibles côté client.
en-têtes de sécurité	En-têtes de réponse HTTP, comme la content security policy et la strict transport security, qui réduisent la surface d'attaque de l'application dans le navigateur.

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 10. Vous pouvez ouvrir vous-même [module-10.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 EnterpriseOps 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 10 – « Architecture de sécurité : identité, SSO, autorisation, audit et déploiement sécurisé » – du cours LearnWisej « Wisej.NET en entreprise : de l'architecture au cloud », 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/azure-docker/specs/module-10.md>

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

1. Projet « EnterpriseOps », 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 : Ajoutez une simulation d'authentification d'entreprise, la correspondance des claims vers les permissions, l'application des rôles par tenant, l'autorisation au niveau des services, la revue du HTML sûr, l'approbation des exports et la journalisation d'audit des actions sensibles.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Conception de la correspondance d'identité
 - Matrice des permissions
 - Implémentation du service de permissions
 - Écran du journal d'audit
 - Checklist de durcissement de la sécurité
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 :

- Où l'autorisation est-elle appliquée?
- Quels éléments de l'UI affichent du HTML?
- Un bouton activé peut-il contourner les règles du service?

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 :

- ✓ Où l'autorisation est-elle appliquée?
- ✓ Quels éléments de l'UI affichent du HTML?
- ✓ Un bouton activé peut-il contourner les règles du service?

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.



Nous l'avons déjà construit — comparez votre version à la nôtre

L'implémentation de référence du Module 10 se trouve dans le dépôt public des exemples. Construisez d'abord la vôtre, puis ouvrez la nôtre : ce sont les différences qui sont intéressantes.



[Ouvrir l'exemple du Module 10 sur GitHub](#)

Ou clonez l'ensemble avec `git clone https://github.com/iceteagroup/LearnWisej-Samples` et ouvrez `Enterprise Wisej.NET Course/Module 10`.



Notre exemple est un point de départ, pas un produit fini

Il a été construit de la même manière que vous construisez le vôtre — par prompt, puis corrigé — et il s'arrête volontairement là où s'arrête le module. Attendez-vous à des éléments provisoires là où irait du code de production, à une gestion des erreurs minimale et à l'absence de tests. Voyez-y une invitation : utilisez ce que vous avez appris dans ce module, avec l'assistant, pour aller plus loin. Améliorer du code que vous n'avez pas écrit, c'est précisément la compétence que ce cours enseigne.

Trois prompts pour aller plus loin

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

- ✓ **Prouvez que ça fonctionne vraiment.** *"Passe en revue ce projet livrable par livrable par rapport à la spécification du module sur <https://learnwisej.com/assets/courses/azure-docker/specs/module-10.md> et montre-moi la preuve que chacun est réellement terminé — le chemin dans le code, où cliquer et ce que je dois voir. Là où un livrable n'est que partiellement satisfait, dis-le clairement au lieu de le défendre. Ensuite, ajoute ce qui manque."*
- ✓ **Relisez-le comme une pull request.** *"Fais-moi parcourir ce code EnterpriseOps 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 EnterpriseOps 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."*



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.