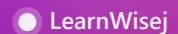


Projet de synthèse — application Mini Helpdesk

Le projet de synthèse · Code lisible et revue par les pairs



Le projet de synthèse

Ce que doit contenir le mini-helpdesk du projet de synthèse — navigation, CRUD, boîtes de dialogue, validation, finition du thème — et comment ses éléments s'articulent.



Objectif de la leçon

Comprendre le projet de synthèse : une petite **application de helpdesk/tickets** Wisej.NET qui réunit tout le contenu du cours — navigation, CRUD, boîtes de dialogue, validation, finition du thème, notes d'architecture et liste de contrôle de déploiement.

1. Ce que le projet de synthèse doit démontrer

- L'application possède un shell clair avec une navigation à gauche et des écrans distincts.
- Les tickets peuvent être consultés, créés, modifiés et supprimés à l'aide d'une `DataGridView` et de boîtes de dialogue.
- La validation est proche de l'interface, mais les règles métier réutilisables se trouvent en dehors d'événements de boutons pris au hasard.
- L'application a un thème cohérent, des espacements soignés, des libellés lisibles et des commandes placées de façon prévisible.
- Le projet comprend des notes d'architecture, une liste de contrôle pour la revue de code par les pairs, les prochaines étapes et des notes de déploiement.

2. Liste de contrôle des fonctionnalités du projet de synthèse

Domaine fonctionnel	Preuves attendues
Shell de navigation	Dashboard, Tickets, Architecture, Code Review, Deployment et Next Steps s'ouvrent depuis une même navigation à gauche cohérente.
CRUD des tickets	Créer, modifier et supprimer des tickets ; une DataGridView se rafraîchit après les modifications.
Boîtes de dialogue & validation	Les boîtes de dialogue Create/Edit Ticket valident les champs obligatoires et affichent un retour clair en cas de réussite ou d'erreur.
Couche de service	<code>TicketService</code> gère les opérations sur les tickets au lieu de mettre toute la logique dans les événements d'interface.
Finition de l'interface	Thème appliqué, espacements nettoyés, libellés lisibles, boutons regroupés de façon cohérente.
Préparation au déploiement	La liste de contrôle de mise en production couvre la configuration, la sécurité, la journalisation, les thèmes/ressources statiques et les notes sur la cible.

3. Architecture et flux de données

Le projet de synthèse relie toute l'application. Vous devez être capable d'expliquer comment les données la traversent de bout en bout.

```
Program.cs      → démarre Window1
Window1.cs      → shell, navigation, pages
Ticket.cs       → modèle de données du ticket
TicketDialog    → formulaire de création/modification + validation
TicketService   → logique CRUD et repository simulé
DataGridView    → affiche les tickets
Thème           → interface soignée
Déploiement     → liste de contrôle de mise en production
```

Le flux : l'utilisateur clique sur Create/Edit, la boîte de dialogue s'ouvre, la validation s'exécute, `TicketService` met à jour la liste des tickets, la `DataGridView` se rafraîchit, et l'état ou les notes de déploiement consignent ce qui a changé.



Vous êtes prêt pour la leçon suivante

Vous savez ce que le projet de synthèse doit contenir et comment ses éléments s'articulent. Ensuite : les habitudes de développeur qui le gardent lisible, et les points de la revue par les pairs qui servent à l'évaluer.

Code lisible et revue par les pairs

Les habitudes de développeur qui gardent le projet de synthèse lisible, et les points de la revue de code par les pairs sur lesquels votre projet est évalué.



Objectif de la leçon

Terminer le projet de synthèse avec les habitudes de développeur junior qui rendent une application Wisej.NET maintenable, et savoir ce qu'une revue de code par les pairs examinera avant la remise.

1. Garder le projet de synthèse lisible

Le projet final ne consiste pas seulement à faire fonctionner l'écran — il doit montrer des habitudes qui gardent l'application facile à maintenir.

Habitude	À quoi elle ressemble dans le projet de synthèse
Noms de contrôles clairs	Des noms comme <code>btnCreateTicket</code> , <code>dgvTickets</code> , <code>lblStatus</code> , <code>txtTitle</code> , <code>cmbPriority</code> — pas <code>button1</code> ni <code>label2</code> .
Petits gestionnaires d'événements	Les clics sur les boutons appellent des méthodes utilitaires ou de service ; ils ne cachent pas des centaines de lignes de logique métier.
Validation réutilisable	Un <code>TicketValidator</code> ou une méthode utilitaire peut être testé et réutilisé par les workflows de création/modification.
Rafraîchissement délibéré	Après un enregistrement/une suppression, rafraîchissez délibérément la grille et le message d'état — sans effets de bord déroutants.
Erreurs sans risque	Affichez des messages conviviaux à l'utilisateur ; gardez les détails internes dans les journaux ou notes des développeurs.

2. Les points de la revue de code par les pairs

- **Nommage** : les classes, contrôles et méthodes décrivent-ils clairement leur rôle ?
- **Séparation des responsabilités** : la logique des tickets est-elle dans un service/validateur plutôt que dispersée dans les événements d'interface ?
- **Navigation** : le relecteur peut-il suivre la façon dont chaque page s'ouvre ?
- **Ergonomie** : les boutons sont-ils placés là où les utilisateurs les attendent, avec des libellés et des retours clairs ?
- **Déploiement** : la configuration, la journalisation, les secrets, les thèmes et les notes de version sont-ils vérifiés avant la revue ?



Montrez tout le cours

Le projet de synthèse doit démontrer tout ce que vous avez appris — navigation, CRUD, boîtes de dialogue, validation, thèmes et déploiement — et pas seulement une fonctionnalité isolée.

3. Remise finale suggérée

Remettez un dossier de projet fonctionnel ou un dépôt GitHub, accompagné de captures d'écran des principaux écrans et d'une courte note sur le déploiement et l'état de préparation. L'écran Next Steps doit décrire ce dont une version prête pour la production aurait encore besoin.



Vous êtes prêt pour le lab

Ensuite : regardez la vidéo pas à pas du projet de synthèse, réussissez la vérification finale des connaissances, puis construisez le mini-helpdesk — un shell de navigation, le CRUD des tickets avec boîtes de dialogue et validation via un service, un thème soigné et des notes d'architecture, de revue et de déploiement.