

Architecture Wisej.NET d'entreprise, leadership technique et gouvernance

Guide de la leçon · Guide du lab / de l'examen



Guide de la leçon

Décisions d'architecture, standards de projet, organisation du travail en équipe, jalons de revue de code. Ce que cela signifie pour une équipe Wisej.NET avancée — et comment aborder ce module.



Enjeu de production

Décisions d'architecture, standards de projet, organisation du travail en équipe, jalons de revue de code.

1. Le problème en production

Le travail Wisej.NET avancé ne consiste plus à savoir si un écran peut être construit. Il s'agit de savoir si une équipe peut faire évoluer une grande application pendant des années sans transformer la productivité en dette technique cachée.

2. Concepts clés

Concept	Pourquoi c'est important
structure de solution d'entreprise	Découpe la solution en projets UI, services applicatifs, domaine et infrastructure, afin que les écrans restent légers et que les règles métier aient un seul emplacement.
architecture decision records (ADR)	Documents courts qui consignent le contexte, les options, la décision et les conséquences d'un choix de conception, afin que l'équipe puisse les revoir et y revenir plus tard.
découpages compatibles avec le concepteur	Garde les formulaires et les pages modifiables dans le concepteur tout en déportant le workflow, la validation et l'accès aux données dans des services que le concepteur ne touche jamais.
jalons de revue	Points de contrôle définis où le code, l'architecture et la sécurité sont revus avant la fusion d'une fonctionnalité, pour que la qualité soit imposée plutôt qu'espérée.
conventions d'équipe	Règles partagées de nommage, de dossiers et d'enregistrement des services qui rendent chaque écran familier à tout développeur de l'équipe.
maîtrise de la dette technique	Pratiques telles que les registres de dette, les budgets de refactoring et les règles de dépréciation, qui gardent les raccourcis visibles et planifiés au lieu de les oublier.

3. Déroulé de la leçon

Les développeurs de niveau intermédiaire savent construire une application orientée production. Les développeurs avancés décident de la manière dont l'application survit à la croissance : plus d'écrans, plus d'utilisateurs, plus d'intégrations, plus d'équipes et plus d'environnements de déploiement. L'objectif n'est pas d'imiter une architecture web générique. L'objectif est de préserver l'avantage de Wisej.NET - une productivité C# riche côté serveur - tout en ajoutant assez de discipline pour accompagner le changement à l'échelle de l'entreprise.

Une gouvernance sans bureaucratie

Une équipe Wisej.NET ne devrait pas avoir besoin d'un comité pour ajouter un bouton. Mais elle a besoin de standards sur l'emplacement de la logique de workflow, la façon dont les écrans utilisent les services, l'application de la sécurité, l'abstraction de l'accès aux données et l'ajout des diagnostics. Des architecture decision records légers suffisent lorsqu'ils expliquent la décision, le contexte, les conséquences et la date de revue.

Privilégier le concepteur ne veut pas dire renoncer à l'architecture

Le concepteur de Visual Studio reste précieux, même dans les projets avancés. La frontière ne se situe pas entre UI visuelle et UI codée ; elle se situe entre l'état de l'UI et les décisions applicatives. Les Forms, les Pages et les UserControls peuvent rester modifiables dans le concepteur tandis que l'orchestration, l'autorisation, la validation, la persistance et l'intégration restent dans les services.



Réflexe de production avancé

Ne vous arrêtez pas quand l'écran paraît correct. Expliquez à qui appartient l'état, les implications de sécurité, les chemins d'échec et le comportement en production avant de considérer le module comme terminé.



Objectifs d'apprentissage

À la fin de ce module, vous saurez :

- ✓ Définir les frontières d'architecture d'une grande application Wisej.NET sans la sur-concevoir.
- ✓ Créer des standards de projet pour l'UI, les services, l'accès aux données, les intégrations, la sécurité, les diagnostics et le déploiement.
- ✓ Utiliser des architecture decision records et des jalons de revue pour garder les équipes alignées.
- ✓ Expliquer comment la productivité de Wisej.NET change les responsabilités d'un lead technique.

Guide du lab / de l'examen

Ce que vous allez construire dans le lab pratique, l'approche suggérée et les livrables attendus.



Lab pratique

Créez le socle de la solution EnterpriseOps Command Center. Ajoutez des projets ou des dossiers pour l'UI, les contrôles partagés, le domaine, les services applicatifs, l'accès aux données, les intégrations, la sécurité, les diagnostics, les ressources et le déploiement. Ajoutez un dossier d'Architecture Decision Records et rédigez l'ADR-001 pour la structure de solution retenue.

Approche suggérée

- ✓ Lisez le problème en production et identifiez le risque d'architecture avant de toucher à l'UI.
- ✓ Créez ou mettez à jour d'abord l'encapsulation par un service, de commande, de modèle ou de composant concernée.
- ✓ Construisez l'écran ou le composant Wisej.NET en suivant les standards de projet établis.
- ✓ Ajoutez une démonstration du chemin d'échec, pas seulement du chemin nominal.
- ✓ Consignez ce qui a changé dans le journal d'apprentissage et mettez à jour la checklist concernée.

Livrables attendus

- ✓ Capture d'écran ou arborescence de la structure de la solution et des dossiers
- ✓ ADR-001 sur la décision de structure de la solution
- ✓ Page des standards de codage de l'équipe
- ✓ Exemple de nommage d'un écran de référence et d'encapsulation par un service
- ✓ Checklist de revue de code appliquée au premier écran

Questions de révision

- ✓ Un nouveau développeur senior peut-il trouver le service de workflow d'un écran en moins d'une minute ?
- ✓ L'équipe sait-elle expliquer pourquoi une frontière existe ?
- ✓ Le concepteur reste-t-il utilisable après le découpage de l'architecture ?



Réflexe de production avancé

Ne vous arrêtez pas quand l'écran paraît correct. Expliquez à qui appartient l'état, les implications de sécurité, les chemins d'échec et le comportement en production avant de considérer le module comme terminé.