■■ Architecture .NET et Bonnes Pratiques

Architecture .NET

En développement .NET, on utilise souvent une **architecture en couches** ou la **Clean Architecture**. Le but est de bien organiser le code en séparant les responsabilités :
- **Domaine** : cœur du métier, règles et logique.
- **Application** : définit les cas d'usage.
- **Infrastructure** : base de données, fichiers, services externes.
- **Présentation** : interface utilisateur, API.

Avantage : chaque couche dépend uniquement de celle du dessous, ce qui rend le projet plus propre et maintenable. On peut aussi utiliser une approche **microservices**, qui apporte plus de flexibilité mais plus de complexité (communication entre services, cohérence des données).

■ Bonnes pratiques en .NET

- Injection de dépendances → ne pas créer les objets partout, les injecter.
- Tests unitaires et tests d'intégration → vérifier le fonctionnement isolé et global.
- Logging → journaliser pour comprendre les bugs.
- Séparer les configurations (Dev / Test / Prod) et ne jamais mettre les mots de passe en dur.
- Sécurité avant tout → HTTPS, validation des entrées, éviter les injections SQL.

■ Conventions de nommage en C#

Élément	Convention	Exemple
Classes / Structs / Enums	PascalCase	OrderService, CustomerDetail
Interfaces	I + PascalCase	IRepository, ILogger
Méthodes / Propriétés	PascalCase	CalculateTotal(), CreatedDate
Champs privés	_camelCase ou camelCase	_repository
Variables / Paramètres	camelCase	orderCount, customerName
Constantes	PascalCase	DefaultSize
Namespaces	PascalCase et logique	MyCompany.MyApp.Module