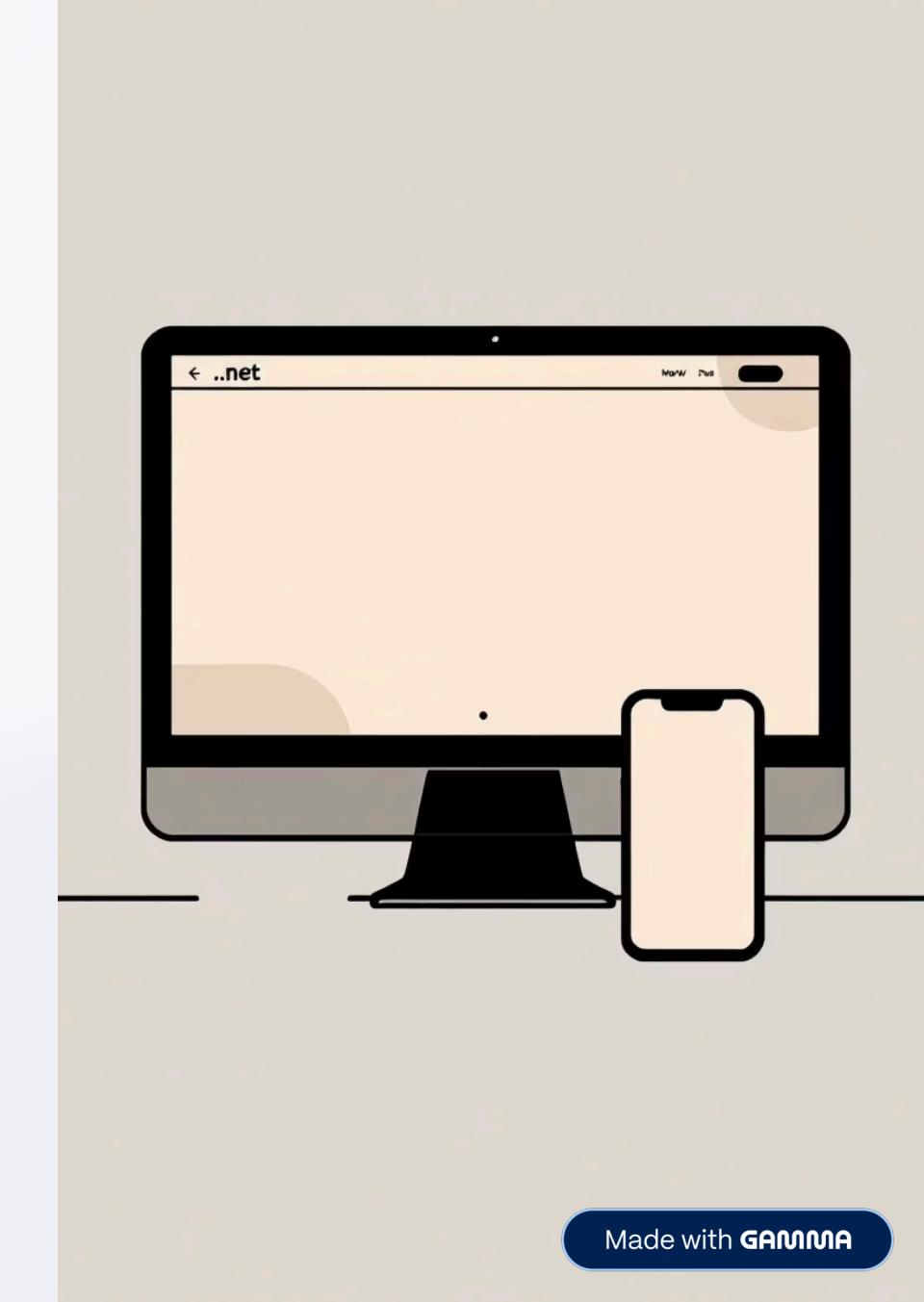


Construire un E-commerce moderne avec ASP.NET Core 8

Une formation pratique pour maîtriser la construction d'applications web robustes et performantes.



CHAPITRE 1

Fondations et Objectifs



Objectifs Clés

Comprendre .NET, l'architecture, le plan d'exécution et les compétences visées.



Public Cible

Développeurs .NET débutants à intermédiaires désirant monter en compétence Full-Stack.



Livrable Attendu

Application e-commerce fonctionnelle, documentée et prête à évoluer (Monolithe MVC/Razor).

- ❑ Approche Full-Stack: Construction complète du Back-end ASP.NET Core et du Front-end Razor à partir de zéro, en privilégiant l'architecture MVC monolithique.

.NET en Bref : Plateforme Moderne et Performante

Le framework .NET se positionne comme un choix incontournable pour le développement d'applications d'entreprise grâce à ses atouts majeurs.

Open Source & Multi-OS

Développez et exécutez vos applications sur **Windows, Linux et macOS**. La communauté active garantit une évolution constante.

Performances Exceptionnelles

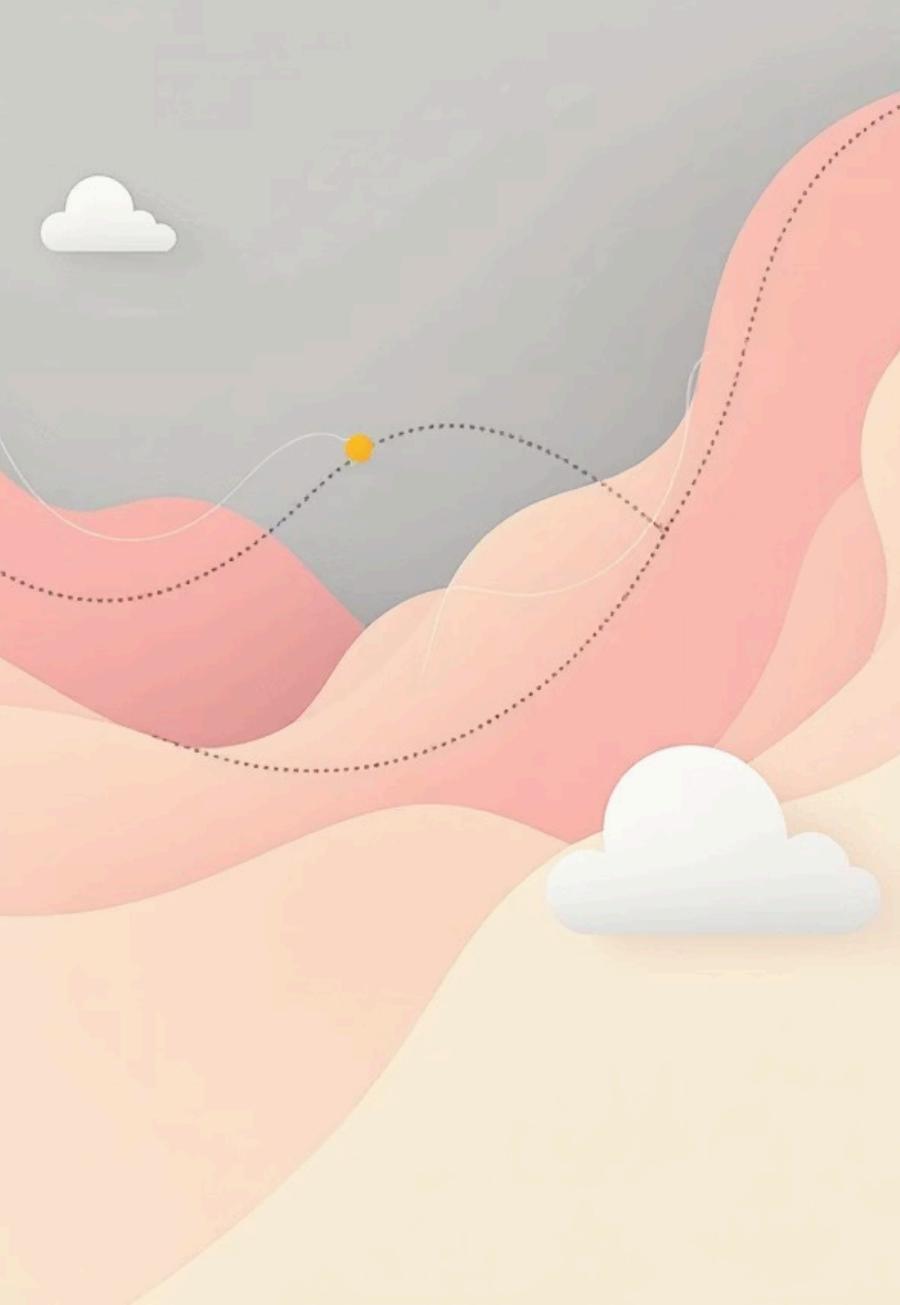
Grâce aux compilateurs JIT (Just-In-Time) et Native AOT, ainsi qu'au serveur web **Kestrel ultra-rapide**.



Écosystème Mûr : Profitez de la richesse de NuGet, de Visual Studio/VS Code et d'un tooling complet.

Versionnement : Nous utilisons **.NET 8 (Long Term Support – LTS)** pour la formation, afin de garantir une base stable, pérenne et conforme aux standards actuels de l'écosystème Microsoft.

ASP.NET Core



ASP.NET Core 9 : Le Cœur du Développement Web

Focus sur les composants essentiels d'ASP.NET Core qui seront utilisés pour construire l'application e-commerce.

1

Modèle MVC et Razor Views

Séparation des préoccupations (Modèle, Vue, Contrôleur) et utilisation des vues Razor pour le rendu HTML dynamique.

2

Injection de Dépendances (DI)

Intégrée nativement via le conteneur IoC : `builder.Services.Add...` pour une gestion des services propre.

3

Gestion de la Configuration

Utilisation standard de `appsettings.json` et du pattern `IOptions` pour une configuration flexible et typée.

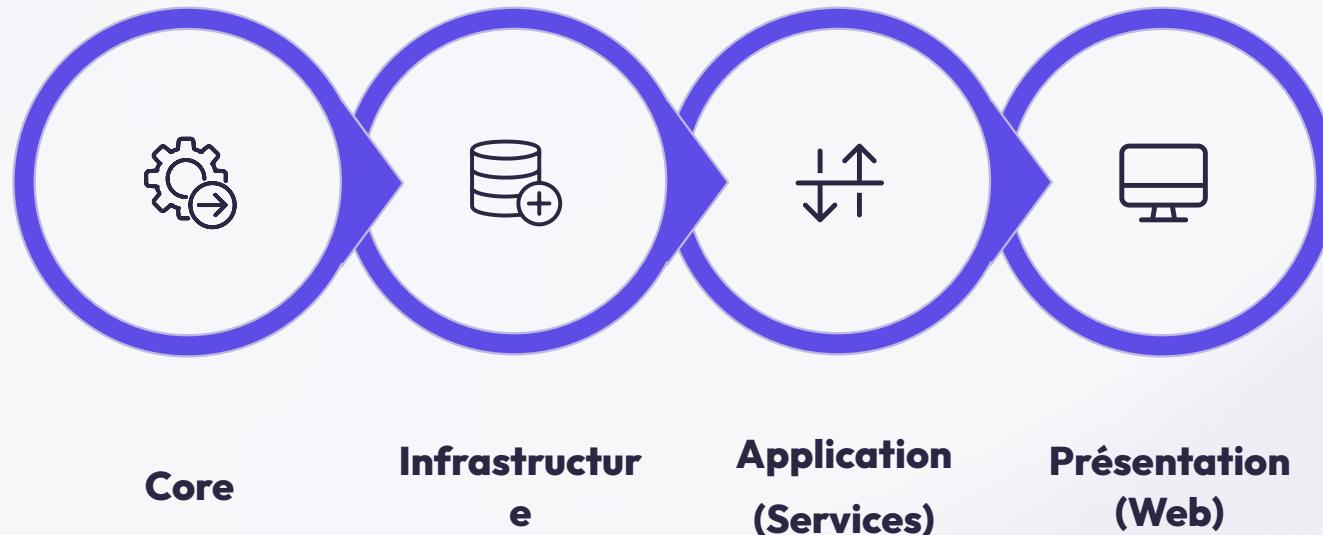
4

Authentification & Identité

Mise en place de l'identité et de l'autorisation avec `AddIdentity` pour gérer les utilisateurs, rôles et tokens.

Architecture du Projet : La "Clean Architecture"

Nous structurerons le projet selon les principes de la Clean Architecture pour garantir la séparation des préoccupations, la testabilité et la maintenabilité.



Formationn_Ecommerce/ (Couche Web)

Point d'entrée HTTP. Contient les **Controllers**, **Views**, et le Mapping d'exposition.



Formationn_Ecommerce.Application/

Contient les **DTOs**, **Services** (Cas d'usage) et la logique applicative.



Formationn_Ecommerce.Core/

Le domaine. Contient les **Entities**, **Interfaces** (Contrats) et les Utilitaires partagés.



Formationn_Ecommerce.Infrastructure/

Implémentation des contrats. Contient **DbContext**, **Repositories** et services externes (Email).

Stack Technique et Packages Clés

Aperçu des technologies et des bibliothèques essentielles qui composent notre stack e-commerce.



EF Core + SQL Server

Gestion des données via `ApplicationDbContext.cs` et les migrations.



ASP.NET Identity

Authentification et gestion des utilisateurs (`AddIdentity()`).



AutoMapper

Simplification du mapping entre Entités, DTOs et ViewModels.



Mailing Externe

Service d'envoi d'emails (ex: via **MailKit/MimeKit**) pour l'activation ou la réinitialisation de mot de passe.

Expérience Utilisateur (Front-end)

- **Bootstrap 5:** Pour un layout responsive et des composants UI modernes.
- **Toastr & TempData:** Notifications utilisateur pour les actions réussies ou les erreurs.
- **DataTables:** Tables interactives avec tri, pagination et recherche pour l'administration.
- **Tag Helpers:** Simplification de la création de formulaires et de la validation côté client/serveur.

Domaines Fonctionnels Couverts par le Projet

Le projet couvre les fonctionnalités essentielles d'une application e-commerce, de la gestion du catalogue à la finalisation de la commande.



Catalogue

Gestion complète des **Produits & Catégories** (CRUD), incluant l'upload sécurisé des images (via IFileHelper).



Promotions

Implémentation des **Coupons** pour appliquer des remises au panier par code unique.



Panier et Commande

Logique complète de gestion du **Panier** (ajout/suppression, totaux) et du processus de **Checkout** (OrderHeader/Details).



Sécurité

Flux d'**Authentification** (Register/Login/Logout) avec gestion d'emails pour la confirmation et la réinitialisation.

Flux Applicatif : De la Requête à l'Affichage

Comprendre le chemin de la donnée et des responsabilités à travers les différentes couches de l'architecture.



1. Requête (Controller MVC)

Le `ProductController.cs` reçoit la requête utilisateur.



2. Logique Métier (Service Application)

Le `ProductServices` exécute la logique, valide les données et effectue le mapping DTOs.



3. Persistance (Repository)

Le `ProductRepository` utilise EF Core pour interagir avec **SQL Server**.



4. Affichage (Razor Views)

Les données sont mappées en `ViewModels`, puis affichées par la vue Razor correspondante (ex: `Views/Product/Index.cshtml`).

Feuille de Route et Compétences

Étapes de Réalisation du Projet

Une progression modulaire pour construire l'application e-commerce étape par étape.



Setup Initial

Création de la solution, des projets, configuration de la DI, AutoMapper et la configuration de base.



Couche Données & Auth

ApplicationContext, Migrations EF Core, mise en place de l'Identity et des vues d'Authentification.



Intégration Front-end

Construction du layout Razor (_Layout.cshtml), intégration de Bootstrap, Toastr et DataTables.



Développement des Domaines

Développement séquentiel des domaines fonctionnels: Produit/Catégorie → Panier → Commande → Coupon.



Finalisation & Intégrations

Ajout de l'intégration Email SMTP, finalisation des validations et des messages utilisateurs via TempData.

Valeur Ajoutée et Compétences Clés Maîtrisées

Ce projet est un catalyseur d'employabilité en fournissant une expérience concrète sur les pratiques professionnelles du développement .NET.



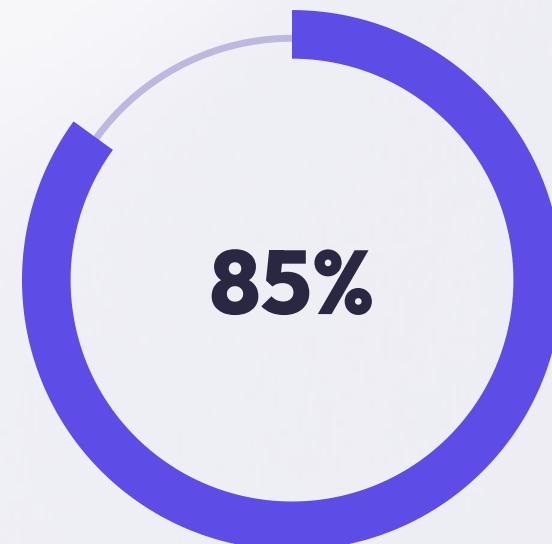
Architecture Propre

Application de la Clean Architecture garantissant la testabilité et la maintenabilité.



Patterns Fondamentaux

Maîtrise des Design Patterns essentiels (Repository, DTO, DI, Options) largement demandés.



Compétences Marché

Expertise concrète sur ASP.NET Core, EF Core, Identity et les intégrations externes sécurisées.



Le livrable final constitue un **portefeuille crédible** pour démontrer vos capacités techniques lors des entretiens d'embauche.