# Cahier de Conception et Diagrammes UML

## Table des Matières

1. [Introduction](#introduction)
2. [Cahier de Conception](#cahier-de-conception)
   1. [Architecture Générale](#architecture-générale)
   2. [Choix Techniques](#choix-techniques)
   3. [Logique Métier Principale](#logique-métier-principale)
3. [Diagramme de Classes UML](#diagramme-de-classes-uml)
   1. [Explication du diagramme de classes](#explication-du-diagramme-de-classes)
4. [Diagrammes de Cas d’Utilisation](#diagrammes-de-cas-dutilisation)
   1. [Explication des cas d’utilisation](#explication-des-cas-dutilisation)
5. [Diagrammes de Séquence UML](#diagrammes-de-séquence-uml)
   1. [Commande d’œuvre](#sequence-commande-oeuvre)
   2. [Gestion d’une œuvre par l’admin](#sequence-gestion-oeuvre-admin)
6. [Conclusion](#conclusion)

## Introduction

Ce document présente la conception détaillée de l’application ArtGallery, incluant les choix d’architecture, la logique métier, ainsi que les principaux diagrammes UML (classes, cas d’utilisation, séquences) pour illustrer la structure et le fonctionnement du système.

## Cahier de Conception

### Architecture Générale

L’application ArtGallery est structurée selon le modèle MVC (Modèle-Vue-Contrôleur) d’ASP.NET Core. Cette architecture sépare clairement la logique métier (Modèles), la gestion des interactions utilisateur (Contrôleurs) et l’affichage (Vues/Razor).

* **Modèles** : Représentent les entités principales du domaine (œuvres, expositions, catégories, utilisateurs, commandes, panier, livre d’or, etc.) et gèrent la logique métier ainsi que la validation des données.
* **Vues** : Les fichiers Razor (.cshtml) affichent les données et fournissent l’interface utilisateur en français, tout en utilisant des routes et noms d’action en anglais pour garantir la cohérence avec les contrôleurs.
* **Contrôleurs** : Gèrent les requêtes HTTP, orchestrent la logique métier et sélectionnent les vues appropriées. Chaque entité principale possède son propre contrôleur (ArtworkController, ExhibitionController, CategoryController, etc.).
* **Données** : L’accès aux données est centralisé via Entity Framework Core, avec un contexte (DbContext) qui expose les DbSet pour chaque entité.

L’organisation des couches permet une maintenance facilitée, une évolutivité et une séparation claire des responsabilités.

### Choix Techniques

* **Framework** : ASP.NET Core MVC (C#)
* **Base de données** : SQL Server (peut être localdb ou une instance distante)
* **ORM** : Entity Framework Core, utilisé pour la gestion des entités, des migrations et des requêtes LINQ
* **Authentification & Sécurité** : Système d’identité ASP.NET Core Identity pour la gestion des utilisateurs, des rôles (Admin, Utilisateur), du chiffrement des mots de passe et de la sécurisation des accès aux ressources sensibles
* **Front-end** : Razor Views avec Bootstrap pour le responsive design, HTML5, CSS3
* **Validation** : Validation côté serveur (DataAnnotations) et côté client (jQuery Validation)
* **Gestion des ressources statiques** : wwwroot (images, CSS, JS)
* **Internationalisation** : Interface utilisateur en français, noms de routes et actions en anglais pour la cohérence technique
* **Déploiement** : Compatible avec IIS, Azure, ou tout autre hébergeur supportant .NET Core

Ces choix assurent robustesse, sécurité, et facilité de développement/maintenance.

### Logique Métier Principale

L’application gère plusieurs processus métier essentiels :

* **Gestion des œuvres d’art** : Ajout, modification, suppression, consultation des œuvres, avec gestion des images et des catégories associées.
* **Gestion des expositions** : Création, édition, suppression et affichage des expositions, avec dates, lieux et œuvres associées.
* **Gestion des catégories** : Organisation des œuvres par catégories, permettant la navigation et le filtrage.
* **Panier d’achat et commandes** : Ajout/retrait d’œuvres au panier, validation du panier, création et gestion des commandes par les utilisateurs, suivi du statut des commandes par les administrateurs.
* **Livre d’or** : Les visiteurs peuvent laisser un message, qui est affiché publiquement après validation.
* **Liens externes** : Gestion de liens vers des ressources partenaires ou externes.
* **Sécurité et gestion des accès** : Rôles différenciés (Admin/Utilisateur), restrictions sur certaines opérations (ex : gestion des œuvres réservée aux admins).

**Règles de gestion principales** : - Un utilisateur doit être authentifié pour commander ou gérer ses commandes. - Seuls les administrateurs peuvent gérer le catalogue d’œuvres, les expositions, les catégories, les liens externes et modérer le livre d’or. - Les routes sont en anglais, mais l’interface utilisateur reste en français pour une meilleure expérience utilisateur.

## Diagramme de Classes UML

### IMG_256

### Explication du diagramme de classes

Ce diagramme de classes représente les principales entités de la base de données ArtGallery et leurs relations : utilisateurs, œuvres, catégories, clients, commandes, détails de commande, expositions, images, liens externes, etc. Les cardinalités sont indiquées par les associations (par exemple, un utilisateur peut avoir plusieurs clients, un client plusieurs commandes, etc.).

### Explication du diagramme de classes

Le diagramme de classes ci-dessus modélise la structure principale de la base de données et du domaine métier de l’application ArtGallery :

* **User** : représente un utilisateur du système (artiste/admin ou client). Un utilisateur peut être lié à plusieurs clients (notion de compte lié à une personne physique).
* **Customer** : un client est une personne qui peut passer des commandes. Il est lié à un User (compte d’accès).
* **Order** : une commande passée par un client, contenant la date, le montant total, le statut, l’adresse de livraison, etc. Un client peut avoir plusieurs commandes.
* **OrderDetail** : détail d’une commande (ligne de commande), associant une œuvre à une commande, avec la quantité et le prix au moment de l’achat.
* **Artwork** : une œuvre d’art, avec titre, description, prix, technique, image principale, etc. Une œuvre appartient à une catégorie et peut être liée à plusieurs commandes (via OrderDetail), à des images supplémentaires (ArtworkImage), à des expositions (ExhibitionArtwork).
* **Category** : catégorie d’œuvres (peinture, sculpture, etc.), chaque œuvre appartient à une catégorie.
* **ArtworkImage** : images supplémentaires pour une œuvre.
* **Exhibition** : exposition d’œuvres, avec lieu, dates, description.
* **ExhibitionArtwork** : table de liaison entre expositions et œuvres (une œuvre peut être exposée plusieurs fois, une exposition contient plusieurs œuvres).
* **GoldenBookEntry** : messages du livre d’or laissés par les visiteurs ou clients.
* **ExternalLink** : liens externes (blogs, partenaires, expositions, etc.).
* **Biography** : biographie de l’artiste.

**Cardinalités principales :** - Un utilisateur peut avoir plusieurs clients (User 1..\* Customer). - Un client peut passer plusieurs commandes (Customer 1..\* Order). - Une commande a plusieurs lignes (Order 1..\* OrderDetail). - Une œuvre peut être dans plusieurs commandes (Artwork 1..\* OrderDetail). - Une œuvre appartient à une catégorie (Artwork *..1 Category). - Une œuvre peut avoir plusieurs images (Artwork 1..* ArtworkImage). - Une œuvre peut être exposée dans plusieurs expositions (Artwork *..* Exhibition via ExhibitionArtwork). - Une exposition contient plusieurs œuvres (Exhibition *..* Artwork via ExhibitionArtwork).

Ce modèle assure la cohérence des données et permet d’exprimer les processus clés de la galerie (vente, gestion d’œuvres, expositions, etc.).

#### Tableau récapitulatif des entités et relations

| Élément | Type | Rôle / Description | Relations principales |
| --- | --- | --- | --- |
| User | Classe | Utilisateur du système | 1..\* Customer |
| Customer | Classe | Client, lié à un User | 1..\* Order, 1 User |
| Order | Classe | Commande passée par un client | 1..\* OrderDetail, 1 Customer |
| OrderDetail | Classe | Ligne de commande (œuvre, quantité, prix) | 1 Order, 1 Artwork |
| Artwork | Classe | Œuvre d’art | 1 Category, \* ArtworkImage, \* ExhibitionArtwork, \* OrderDetail |
| Category | Classe | Catégorie d’œuvres | \* Artwork |
| ArtworkImage | Classe | Image supplémentaire d’une œuvre | 1 Artwork |
| Exhibition | Classe | Exposition d’œuvres | \* ExhibitionArtwork |
| ExhibitionArtwork | Classe | Liaison exposition/œuvre | 1 Exhibition, 1 Artwork |
| GoldenBookEntry | Classe | Message livre d’or | - |
| ExternalLink | Classe | Lien externe | - |
| Biography | Classe | Biographie de l’artiste | - |

## Diagrammes de Cas d’Utilisation

### IMG_256

### Explication des cas d’utilisation

Ce diagramme illustre les interactions majeures entre les acteurs (visiteur, client, admin/artiste) et le système. Il détaille les possibilités offertes selon le rôle : consultation, gestion, achat, etc.

### Explication des cas d’utilisation

Le diagramme de cas d’utilisation présente les principaux scénarios d’interaction entre les acteurs et le système :

* **Visiteur** : peut consulter la galerie, créer un compte, se connecter, laisser un message dans le livre d’or.
* **Client** : après authentification, peut ajouter des œuvres au panier, commander des œuvres, consulter l’historique de ses commandes.
* **Admin/Artiste** : gère l’ensemble du contenu (œuvres, expositions, commandes, liens externes), accède à des fonctionnalités avancées de gestion et de modération.

Chaque cas d’utilisation correspond à une fonctionnalité majeure du système. Les flèches indiquent quels rôles ont accès à quelles fonctionnalités. Ce diagramme aide à visualiser la couverture fonctionnelle et à vérifier que tous les besoins utilisateurs sont pris en compte.

#### Tableau des acteurs et cas d’utilisation

| Acteur / Élément | Type | Description / Action principale | Interactions principales |
| --- | --- | --- | --- |
| Visiteur | Acteur | Parcourt la galerie, crée un compte, laisse un message | Consulter galerie, Créer compte, Livre d’or |
| Client | Acteur | Achète, consulte commandes, utilise panier | Ajouter au panier, Commander, Historique |
| Admin/Artiste | Acteur | Gère tout le contenu et la modération | Gérer œuvres, expositions, commandes |
| Galerie | Système | Application ArtGallery | Fournit toutes les fonctionnalités |
| Cas d’utilisation | Cas | Fonctionnalités accessibles selon le rôle | Voir diagramme pour les liens |

## Diagrammes de Séquence UML

### IMG_256

### Commande d’œuvre

Cette séquence montre le processus complet de commande d’une œuvre par un client :

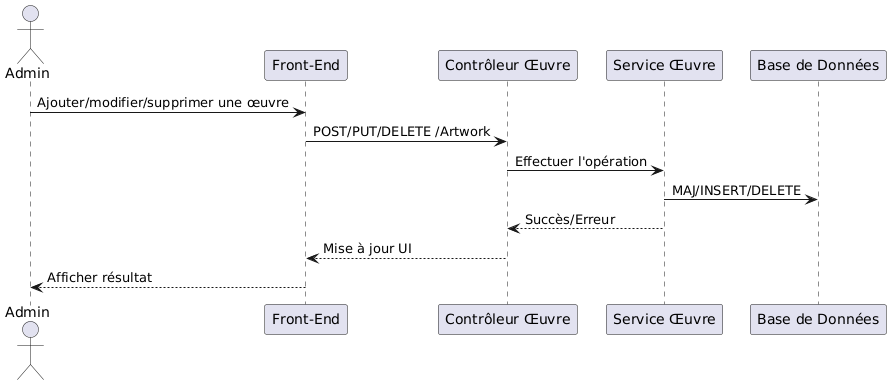
1. **Sélection de l’œuvre** : le client parcourt la galerie et sélectionne une œuvre.
2. **Ajout au panier** : l’œuvre est ajoutée au panier côté front-end.
3. **Validation du panier** : le client valide son panier pour passer commande.
4. **Création de commande** : le front-end envoie la demande de création de commande au contrôleur.
5. **Traitement métier** : le contrôleur appelle le service qui gère la logique de création de commande (insertion en base).
6. **Confirmation** : en cas de succès, le client est redirigé vers une page de confirmation.

Ce diagramme permet de visualiser le flux d’information entre les différentes couches (UI, contrôleur, service, base de données) lors d’un achat.

#### Tableau des participants et interactions (Commande d’œuvre)

| Participant | Type | Rôle / Action principale | Interactions principales |
| --- | --- | --- | --- |
| Client | Acteur | Sélectionne et commande une œuvre | Interagit avec le front-end |
| Front-End (FE) | Interface | Gère l’UI, panier, validation | Dialogue avec Client et OrderController |
| OrderController | Contrôleur | Reçoit la commande, appelle le service | Dialogue avec FE et OrderService |
| OrderService | Service | Gère la logique de création de commande | Dialogue avec OrderController et DB |
| Base de Données (DB) | Base de données | Stocke commandes et détails | Dialogue avec OrderService |

### Gestion d’une œuvre par l’admin



Cette séquence décrit le processus d’ajout, de modification ou de suppression d’une œuvre par l’administrateur :

1. **Action de l’admin** : l’admin utilise l’interface pour créer, modifier ou supprimer une œuvre.
2. **Appel du contrôleur** : la requête est envoyée au contrôleur côté serveur.
3. **Traitement métier** : le contrôleur délègue l’opération au service métier, qui effectue les vérifications et la gestion des données.
4. **Interaction avec la base** : le service effectue les opérations nécessaires sur la base de données (insertion, mise à jour, suppression).
5. **Retour du résultat** : le résultat (succès ou erreur) remonte jusqu’à l’interface utilisateur.

Ce diagramme illustre la séparation des responsabilités et la chaîne de traitement lors de la gestion des œuvres.

#### Tableau des participants et interactions (Gestion d’œuvre admin)

| Participant | Type | Rôle / Action principale | Interactions principales |
| --- | --- | --- | --- |
| Admin | Acteur | Gère les œuvres (ajout, modif, suppression) | Interagit avec le front-end |
| Front-End (FE) | Interface | Gère l’UI, transmet les actions à l’admin | Dialogue avec Admin et ArtworkController |
| Composant | Type | Rôle | Interactions |
| ———————— | —————— | —————————————————- | ——————————————- |
| ArtworkController | Contrôleur | Reçoit les requêtes, appelle le service | Dialogue avec le Front-End et ArtworkService |
| ArtworkService | Service | Gère la logique métier des œuvres | Dialogue avec ArtworkController et la DB |
| Base de Données (DB) | Base de données | Stocke/modifie/supprime les œuvres | Dialogue avec ArtworkService |

## Conclusion

Ce cahier de conception fournit une vision globale et détaillée de l’architecture, des choix techniques et des principaux flux de l’application ArtGallery. Les diagrammes UML facilitent la compréhension et la communication autour du projet.