ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

HONORIS UNITED UNIVERSITIES

2025 - 2026

4ème Année Ingénierie Informatique et Réseaux 14

Rapport du Projet de Contrôle .NET

Sous le Thème

Application Web de Gestion des Stages des Étudiants

V.1

Réalisé par

Aya EL HADDAJ

TABLE DES MATIERES

TABLE DES FIGURES

1.INTRODUCTION

1.1. Contexte général

La gestion des stages représente un enjeu majeur pour les établissements d'enseignement supérieur et les entreprises. Le processus traditionnel de gestion des stages, souvent basé sur des échanges d'emails et des documents papier, présente plusieurs limitations : difficulté de suivi, perte d'information, manque de traçabilité et temps de traitement important.

Face à ces contraintes, la digitalisation du processus de gestion des stages s'impose comme une solution incontournable. Une application web centralisée permet d'optimiser le workflow complet, depuis la publication des offres jusqu'au dépôt des rapports de stage, tout en offrant une interface adaptée à chaque acteur du processus.

1.2. Objectif du projet

1.2.1. Objectif principal

L'objectif principal de ce projet est de développer une application web complète de gestion des stages qui automatise et facilite l'ensemble du processus pour les trois acteurs principaux : les étudiants, les entreprises et l'administration.

1.2.2. Objectifs spécifiques

Les objectifs spécifiques du projet sont les suivants :

- **Automatiser la publication et la consultation des offres de stage** : permettre aux entreprises de publier leurs offres et aux étudiants de les consulter facilement.
- **Digitaliser le processus de candidature** : offrir aux étudiants la possibilité de postuler en ligne avec téléversement de CV.
- **Faciliter la gestion des candidatures** : permettre aux entreprises d'accepter ou de refuser les candidatures de manière simple et traçable.
- **Gérer les conventions de stage** : permettre à l'administration de créer et de gérer les conventions avec upload des documents signés.
- **Centraliser les rapports de stage** : offrir un espace de dépôt sécurisé pour les rapports avec possibilité de téléchargement.
- **Assurer la sécurité et la confidentialité** : implémenter un système d'authentification et d'autorisation basé sur les rôles pour protéger les données sensibles.

1.3. Périmètre du projet

Le projet couvre l'ensemble du processus de gestion des stages à travers les fonctionnalités suivantes :

Authentification et gestion des utilisateurs :

- Système d'authentification multi-rôles (Administrateur, Étudiant, Entreprise)
- Gestion des profils utilisateurs avec modification des informations personnelles

Gestion des offres de stage :

- Publication, modification et suppression des offres par les entreprises
- Consultation des offres par les étudiants
- Recherche et filtrage avancés des offres (par secteur, durée, entreprise)

Gestion des candidatures :

- Soumission de candidatures avec téléversement de CV
- Suivi en temps réel du statut des candidatures
- Workflow d'acceptation/refus par les entreprises
- Notifications par email à chaque changement de statut

Gestion des conventions de stage :

- Création de conventions pour les candidatures acceptées
- Téléversement de documents PDF signés
- Calcul automatique du statut des stages (À venir, En cours, Terminé)
- Téléchargement des conventions par les parties concernées

Gestion des rapports de stage :

- Dépôt de rapports par les étudiants avec upload de fichiers PDF
- Consultation et téléchargement des rapports
- Lien automatique entre rapport et convention

Fonctionnalités avancées :

- Tableau de bord avec statistiques et graphiques pour l'administration
- Notifications automatiques par email pour les événements importants
- Export des données en formats Excel et CSV

- Interface utilisateur responsive et moderne avec Bootstrap 5

Cette application web offre ainsi une solution complète et intégrée pour la digitalisation du processus de gestion des stages.

2. ANALYSE ET CONCEPTION

2.1. Analyse des besoins

2.1.1. Identification des acteurs

Le système de gestion des stages implique trois acteurs principaux, chacun ayant des besoins et des responsabilités spécifiques :

Administrateur

- Rôle : Superviser l'ensemble du système et gérer les conventions de stage
- Responsabilités : Création et gestion des conventions, supervision des rapports, gestion globale du système

Étudiant

- Rôle : Rechercher des opportunités de stage et suivre son parcours
- Responsabilités : Consulter les offres, postuler avec CV, suivre ses candidatures, déposer son rapport de stage

Entreprise

- Rôle : Publier des offres et recruter des stagiaires
- Responsabilités : Créer des offres de stage, consulter les candidatures reçues, accepter ou refuser les candidats

2.1.2. Besoins fonctionnels par acteur

Besoins de l'Administrateur :

- S'authentifier au système
- Créer et gérer les conventions de stage
- Uploader les conventions signées en format PDF
- Consulter tous les rapports de stage déposés
- Superviser l'ensemble du processus de gestion des stages
- Accéder aux statistiques et tableaux de bord

Besoins de l'Étudiant :

- S'inscrire et s'authentifier
- Gérer son profil (nom, prénom, filière, niveau, etc.)
- Consulter les offres de stage disponibles

- Rechercher et filtrer les offres selon ses critères
- Postuler aux offres avec téléversement de CV
- Suivre le statut de ses candidatures
- Consulter ses conventions de stage
- Déposer son rapport de stage en format PDF
- Recevoir des notifications par email

Besoins de l'Entreprise :

- S'inscrire et s'authentifier
- Gérer son profil (nom, secteur, adresse, contact, etc.)
- Créer, modifier et supprimer des offres de stage
- Consulter les candidatures reçues
- Télécharger les CV des candidats
- Accepter ou refuser les candidatures
- Consulter les conventions liées à ses offres
- Recevoir des notifications par email

2.1.3. Besoins non fonctionnels

Sécurité :

- Authentification obligatoire pour accéder au système
- Autorisation basée sur les rôles (RBAC)
- Protection des données personnelles
- Sécurisation des fichiers uploadés

Performance :

- Temps de réponse rapide pour les requêtes
- Gestion optimisée des fichiers PDF
- Requêtes de base de données optimisées avec LINQ

Ergonomie :

- Interface utilisateur intuitive et moderne
- Design responsive adapté à tous les appareils
- Navigation simple et cohérente

Fiabilité :

- Gestion des erreurs et exceptions
- Validation des données côté client et serveur
- Sauvegarde automatique des données

2.2. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation illustre les interactions entre les acteurs et le système. Il présente une vue d'ensemble des fonctionnalités organisées en cas d'utilisation généraux et leurs extensions.

2.2.1. Cas d'utilisation principaux

Le système propose six cas d'utilisation généraux :

1. **S'authentifier :** Connexion sécurisée au système (tous les acteurs)
2. **Gérer son profil :** Consultation et modification des informations personnelles
3. **Gérer les offres de stage :** Publication et gestion des opportunités de stage
4. **Gérer les candidatures :** Soumission et traitement des candidatures
5. **Gérer les conventions :** Création et suivi des conventions de stage
6. **Gérer les rapports :** Dépôt et consultation des rapports de stage

2.2.2. Relations entre cas d'utilisation

Relations d'extension (<<extend>>)

Les actions spécifiques étendent les cas généraux. Par exemple, le cas "Gérer les offres de stage" est étendu par :

- Consulter offres
- Créer offre
- Modifier offre
- Supprimer offre
- Rechercher offres

Relations d'inclusion (<<include>>)

Certaines actions incluent obligatoirement d'autres actions :

- "Postuler" inclut "Uploader CV"
- "Créer convention" inclut "Uploader PDF signé"
- "Déposer rapport" inclut "Uploader rapport PDF"

2.2.3. Répartition des fonctionnalités par acteur

L'Étudiant peut :

- Gérer son profil (consulter, modifier)
- Gérer les offres (consulter, rechercher)
- Gérer les candidatures (consulter, postuler, annuler)
- Gérer les conventions (consulter)
- Gérer les rapports (déposer, consulter, modifier)

L'Entreprise peut :

- Gérer son profil (consulter, modifier)
- Gérer les offres (créer, modifier, supprimer, consulter)
- Gérer les candidatures (consulter, accepter, refuser)
- Gérer les conventions (consulter)

L'Administrateur peut :

- Gérer les utilisateurs (ajouter étudiant/entreprise, modifier, supprimer, rechercher)
- Gérer les conventions (créer, modifier, supprimer, consulter, uploader PDF)
- Gérer les rapports (consulter, supprimer)
- Superviser le système

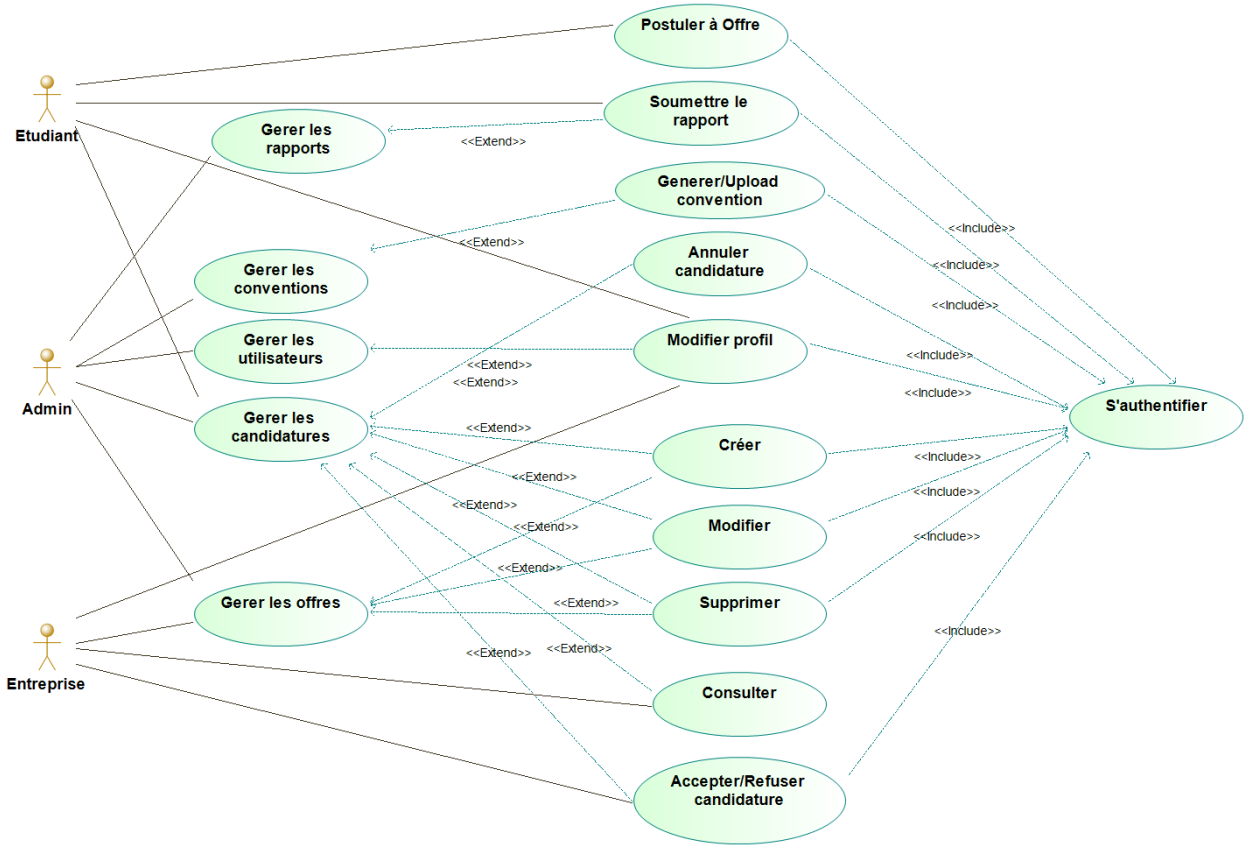

Figure 1:Diagramme de cas d'utilisation

2.3. Conception de la base de données

2.3.1. Diagramme de classes

Le diagramme de classes représente la structure des données du système et les relations entre les différentes entités. Il constitue la base du modèle de données utilisé par l'application.

2.3.2. Description des entités principales

Entité Etudiant

- Attributs : Id, Nom, Prenom, Email, Telephone, Filiere, Niveau
- Relations : Un étudiant peut soumettre plusieurs candidatures (1 à n)
- Rôle : Représente les utilisateurs de type étudiant

Entité Entreprise

- Attributs : Id, Nom, Secteur, Adresse, Telephone, EmailContact
- Relations : Une entreprise peut publier plusieurs offres de stage (1 à n)
- Rôle : Représente les utilisateurs de type entreprise

Entité OffreStage

- Attributs : Id, Titre, Description, DureeMois, DateDebutSouhaitee, DatePublication, EntrepriseId
- Relations : Une offre appartient à une entreprise (n à 1), une offre peut recevoir plusieurs candidatures (1 à n)
- Rôle : Représente les opportunités de stage publiées

Entité Candidature

- Attributs : Id, DateCandidature, Statut, CheminCV, EtudiantId, OffreStageId
- Relations : Une candidature appartient à un étudiant (n à 1) et à une offre (n à 1), une candidature acceptée génère une convention (1 à 1)
- Rôle : Représente les demandes de stage des étudiants
- Statuts possibles : "En attente", "Acceptée", "Refusée"

Entité Convention

- Attributs : Id, DateSignature, DateDebut, DateFin, Statut, CheminFichierPDF, CandidatureId
- Relations : Une convention est liée à une candidature (1 à 1), une convention peut avoir un rapport (1 à 1)

- Rôle : Représente les conventions officielles de stage
- Statuts possibles : "Signée", "En cours", "Terminée"
- Note : Le statut est calculé automatiquement selon les dates

Entité RapportStage

- Attributs : Id, Titre, NomFichier, DateDepot, ConventionId
- Relations : Un rapport est lié à une convention (1 à 1)
- Rôle : Représente les rapports déposés par les étudiants

Entités Identity (ASP.NET Core Identity)

- IdentityUser : Gère l'authentification (Email, Password, UserName)
- IdentityRole : Définit les rôles (Admin, Etudiant, Entreprise)
- Relation : Un utilisateur possède un ou plusieurs rôles

2.3.3. Relations entre entités

Relations principales :

1. **Entreprise ↔ OffreStage** : Une entreprise publie plusieurs offres (1 à n)
2. **OffreStage ↔ Candidature** : Une offre reçoit plusieurs candidatures (1 à n)
3. **Etudiant ↔ Candidature** : Un étudiant soumet plusieurs candidatures (1 à n)
4. **Candidature ↔ Convention** : Une candidature acceptée génère une convention (1 à 1)
5. **Convention ↔ RapportStage** : Une convention contient un rapport (1 à 1)

Associations avec Identity :

- Un IdentityUser (Email) correspond à un Etudiant (relation 1 à 1)
- Un IdentityUser (Email) correspond à une Entreprise (relation 1 à 1)
- Un IdentityUser peut avoir plusieurs IdentityRole (relation n à n)

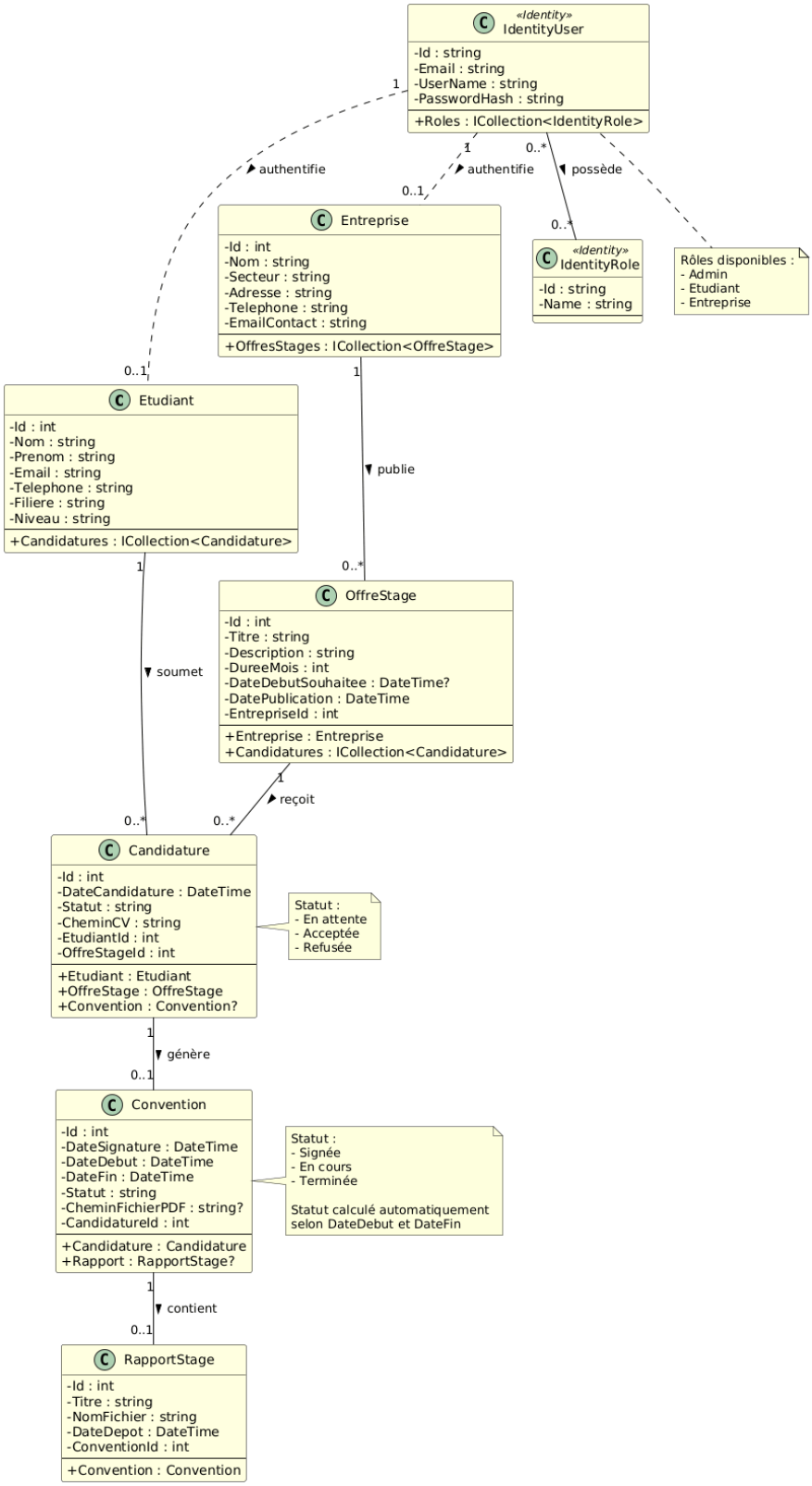

Figure 2: Diagramme de classe

2.4. Schéma de la base de données

2.4.1. Architecture de la base de données

L'application utilise **SQL Server** comme système de gestion de base de données. La base de données est organisée en deux parties distinctes pour séparer les responsabilités :

Base de données d'authentification (AspNetIdentity)

- Gère l'authentification et l'autorisation des utilisateurs
- Tables générées automatiquement par ASP.NET Core Identity
- Stocke les comptes utilisateurs, les rôles et les permissions

Base de données métier (Stages)

- Gère les données fonctionnelles de l'application
- Tables créées via Entity Framework Core (Code First)
- Stocke les entités : Etudiants, Entreprises, Offres, Candidatures, Conventions, Rapports

2.4.2. Base de données d'authentification (AspNetIdentity)

Les tables principales générées par Identity Framework sont :

AspNetUsers

- Stocke les comptes utilisateurs
- Colonnes principales : Id, UserName, Email, PasswordHash, EmailConfirmed, PhoneNumber
- Lien avec les tables métier via l'Email

AspNetRoles

- Définit les rôles disponibles
- Colonnes : Id, Name, NormalizedName
- Rôles créés : Admin, Etudiant, Entreprise

AspNetUserRoles

- Table de liaison entre utilisateurs et rôles (relation n à n)
- Colonnes : UserId, RoleId

Autres tables Identity :

- AspNetUserClaims : Claims personnalisés des utilisateurs

- AspNetUserLogins : Connexions externes (Google, Facebook, etc.)
- AspNetUserTokens : Tokens de sécurité
- AspNetRoleClaims : Claims associés aux rôles

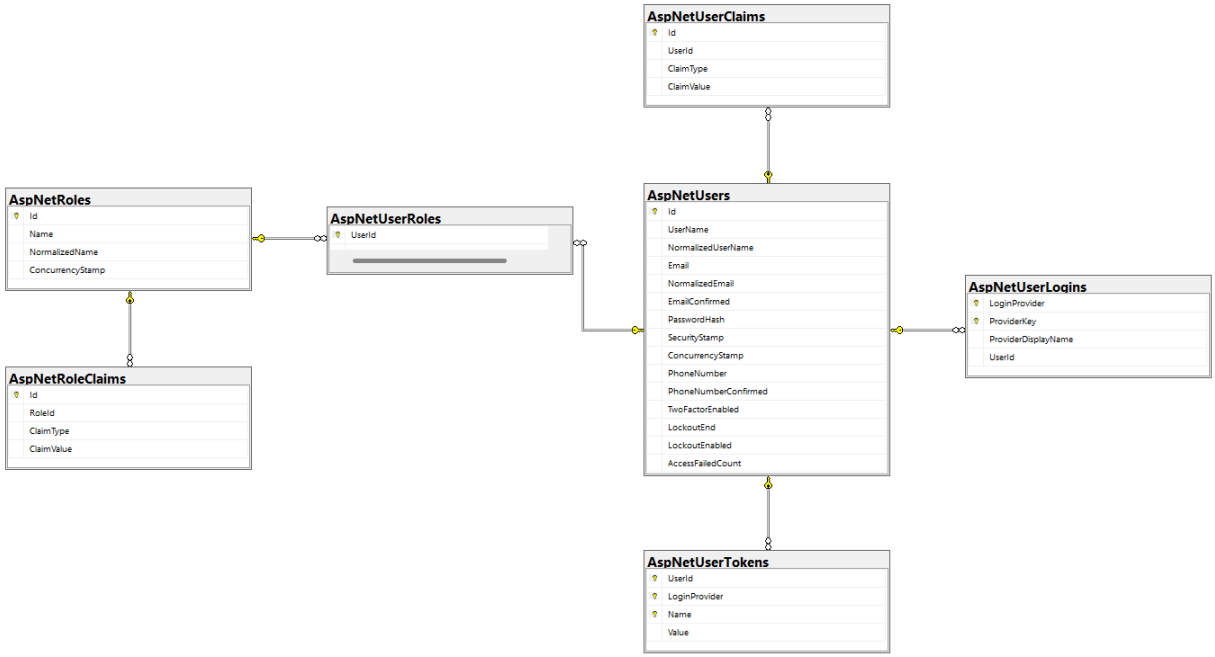

Figure 3:Base de données d'authentification

2.4.3. Base de données métier (Stages)

Les tables métier créées par Entity Framework Core sont :

Etudiants

- Colonnes : Id (PK), Nom, Prenom, Email (UNIQUE), Telephone, Filiere, Niveau
- Clé unique : Email (pour lien avec AspNetUsers)

Entreprises

- Colonnes : Id (PK), Nom, Secteur, Adresse, Telephone, EmailContact (UNIQUE)
- Clé unique : EmailContact (pour lien avec AspNetUsers)

OffresStages

- Colonnes : Id (PK), Titre, Description, DureeMois, DateDebutSouhaitee, DatePublication, EntrepriseId (FK)
- Clé étrangère : EntrepriseId → Entreprises(Id)
- Index : EntrepriseId pour optimiser les requêtes

Candidatures

- Colonnes : Id (PK), DateCandidature, Statut, CheminCV, EtudiantId (FK), OffreStageId (FK)
- Clés étrangères :
  - EtudiantId → Etudiants(Id)
  - OffreStageId → OffresStages(Id)
- Index : EtudiantId, OffreStageId
- Contrainte : Statut IN ('En attente', 'Acceptée', 'Refusée')

Conventions

- Colonnes : Id (PK), DateSignature, DateDebut, DateFin, Statut, CheminFichierPDF, CandidatureId (FK)
- Clé étrangère : CandidatureId → Candidatures(Id) UNIQUE
- Relation 1-1 avec Candidatures
- Contrainte : DateFin > DateDebut

RapportsStages

- Colonnes : Id (PK), Titre, NomFichier, DateDepot, ConventionId (FK)

- Clé étrangère : ConventionId → Conventions(Id) UNIQUE
- Relation 1-1 avec Conventions

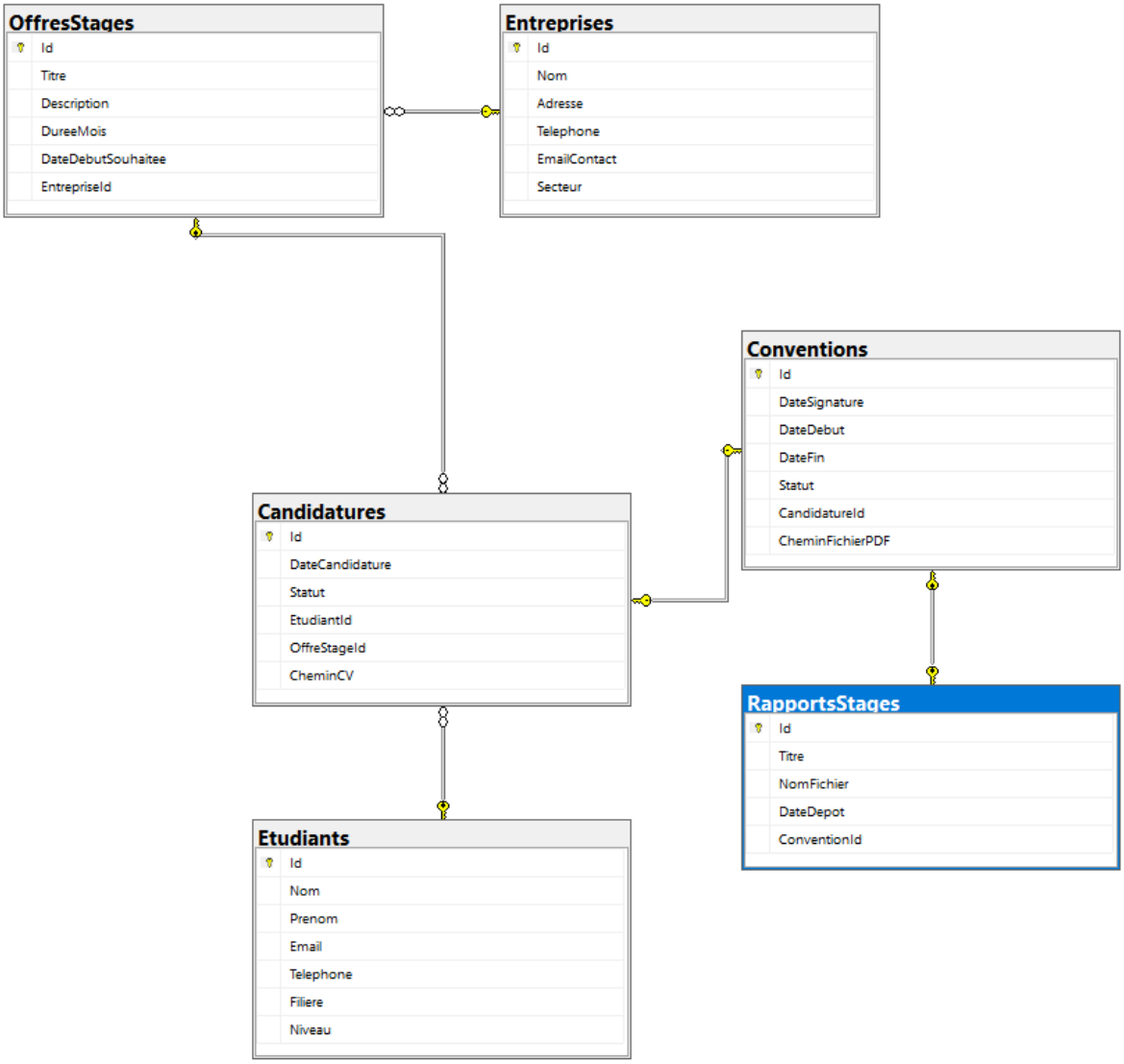

Figure 4: Base de données métiers

2.5. Diagramme de séquence

2.5.1. Processus global du workflow

Le diagramme de séquence illustre le processus complet de gestion d'un stage, de la publication de l'offre jusqu'au dépôt du rapport final. Ce workflow se décompose en cinq phases séquentielles.

2.5.2. Description des phases

Phase 1 : Publication de l'Offre

- L'entreprise se connecte au système
- Elle publie une offre de stage avec titre, description et durée
- Le système enregistre l'offre dans la base de données
- L'offre devient visible pour les étudiants

Phase 2 : Candidature

- L'étudiant consulte les offres disponibles
- Il sélectionne une offre qui l'intéresse
- Il soumet sa candidature en téléversant son CV
- Le système enregistre la candidature avec le statut "En attente"
- L'entreprise reçoit une notification de la nouvelle candidature

Phase 3 : Traitement de la Candidature

- L'entreprise consulte les candidatures reçues
- Elle télécharge et examine le CV de l'étudiant
- Elle décide d'accepter ou de refuser la candidature
- Le système met à jour le statut de la candidature
- L'étudiant reçoit une notification du résultat

Phase 4 : Création de la Convention

- L'administrateur crée une convention pour la candidature acceptée
- Il saisit les dates de début et de fin du stage
- Il téléverse le document PDF de la convention signée
- Le système enregistre la convention et calcule automatiquement son statut
- L'étudiant et l'entreprise reçoivent une notification

Phase 5 : Dépôt du Rapport

- À la fin du stage, l'étudiant accède à sa convention
- Il dépose son rapport de stage en téléversant un fichier PDF
- Le système enregistre le rapport et le lie à la convention
- L'administrateur reçoit une notification du nouveau rapport

2.5.3. Interactions entre acteurs et système

Le diagramme met en évidence :

- Les échanges entre les acteurs et le système
- Les consultations et modifications de la base de données
- Le stockage des fichiers uploadés
- Les notifications automatiques envoyées aux parties concernées
- La progression chronologique du processus

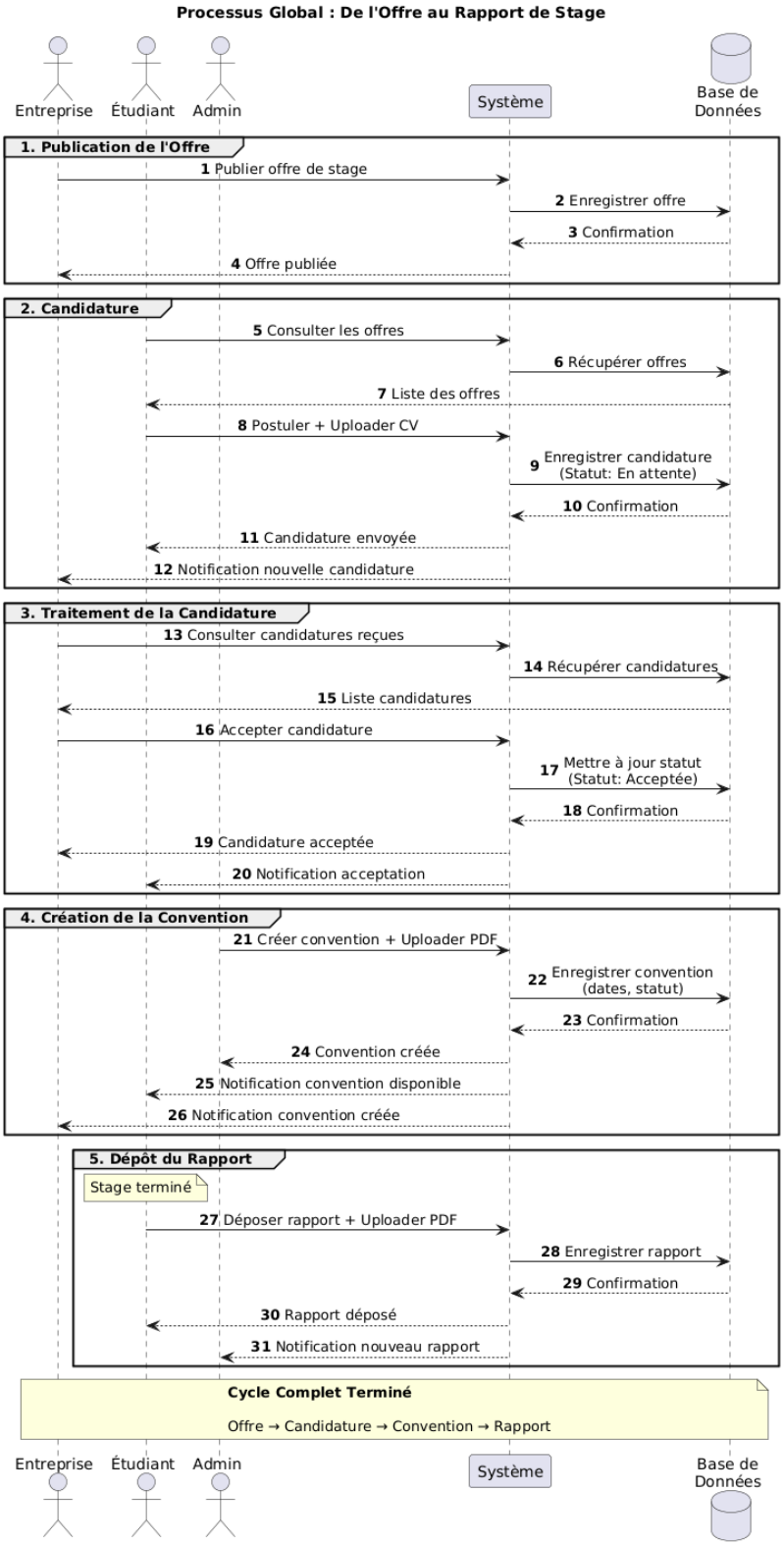

Figure 5: Diagramme de séquence

3. RÉALISATION TECHNIQUE

3.1. Environnement et technologies

3.1.1. Technologies utilisées

Framework principal

- **ASP.NET Core 8.0 MVC** : Framework web moderne de Microsoft pour le développement d'applications web

Base de données et ORM

- **SQL Server** : Système de gestion de base de données relationnelle
- **Entity Framework Core 8.0** : ORM (Object-Relational Mapping) pour l'accès aux données
- **LINQ** : Language Integrated Query pour les requêtes de base de données

Authentification et sécurité

- **ASP.NET Core Identity** : Système d'authentification et de gestion des utilisateurs intégré
- **Role-Based Authorization** : Gestion des permissions par rôles

Interface utilisateur

- **Bootstrap 5** : Framework CSS responsive pour le design
- **Razor Pages** : Moteur de template pour les vues

Outils de développement

- **Visual Studio 2022** : Environnement de développement intégré (IDE)
- **SQL Server Management Studio (SSMS)** : Gestion de la base de données

3.1.2. Justification des choix

ASP.NET Core MVC a été choisi pour :

- Architecture MVC claire et maintenable
- Performance élevée et cross-platform
- Écosystème riche (Identity, EF Core)
- Documentation complète de Microsoft

Entity Framework Core permet :

- Développement Code First (modèle → base de données)

- Migrations automatiques pour la gestion du schéma
- Requêtes LINQ typées et sécurisées

Bootstrap 5 offre :

- Design moderne et professionnel
- Responsive par défaut (mobile, tablette, desktop)
- Composants prêts à l'emploi

3.2. Architecture du projet

3.2.1. Architecture MVC

L'application suit le pattern **Model-View-Controller** qui sépare les responsabilités :

Model (Modèle)

- Contient les entités de données (Etudiant, Entreprise, OffreStage, etc.)
- Définit la structure de la base de données
- Gère la logique métier

View (Vue)

- Fichiers Razor (.cshtml) pour l'interface utilisateur
- Présentation des données à l'utilisateur
- Formulaires de saisie

Controller (Contrôleur)

- Traite les requêtes HTTP
- Interagit avec les modèles via Entity Framework
- Retourne les vues appropriées

3.2.2. Organisation du code

```
GestionStages/
├── Controllers/          # Contrôleurs MVC
├── Models/               # Entités de données
├── Views/                # Vues Razor
│   ├── OffresStages/
│   ├── Candidatures/
│   ├── Conventions/
│   └── RapportStages/
├── Data/                 # DbContext
├── wwwroot/              # Fichiers statiques
│   └── uploads/          # Fichiers uploadés (CV, PDF)
└── Areas/Identity/       # Pages d'authentification
```

3.3. Approches méthodologiques

3.3.1. Approche Code First (Entity Framework Core)

Principe : On définit d'abord les classes C# (entités), puis EF Core génère automatiquement la base de données.

Workflow :

1. Création des classes entités (Models)
2. Configuration du DbContext
3. Création d'une migration : Add-Migration NomMigration
4. Application à la base : Update-Database

Avantages :

- Pas besoin d'écrire du SQL manuellement
- Synchronisation automatique code ↔ base de données
- Historique des modifications via migrations

3.3.2. Approche RBAC (Role-Based Access Control)

Principe : Chaque utilisateur possède un rôle qui détermine ses permissions.

Rôles implémentés :

- **Admin** : Accès complet, gestion des conventions
- **Etudiant** : Consultation offres, candidatures, dépôt rapports
- **Entreprise** : Publication offres, gestion candidatures

Implémentation :

```
[Authorize(Roles = "Admin,Etudiant")]
public async Task<IActionResult> Index() {
    // Filtrage des données selon le rôle
    if (User.IsInRole("Etudiant")) {
        // Afficher uniquement ses données
    }
}
```

3.3.3. Utilisation de LINQ

LINQ (Language Integrated Query) est utilisé pour toutes les requêtes de base de données.

Exemples d'utilisation :

- **Filtrage** : .Where(c => c.Statut == "Acceptée")
- **Chargement de relations** : .Include(r => r.Convention).ThenInclude(c => c.Candidature)
- **Recherche** : .FirstOrDefaultAsync(e => e.Email == userEmail)
- **Projection** : .Select(c => new { c.Id, c.Nom })

Avantages :

- Requêtes typées et vérifiées à la compilation
- SQL généré automatiquement et optimisé
- Code lisible et maintenable

3.4. Fonctionnalités implémentées

3.4.1. Module Authentification

Inscription :

- Choix du type de compte (Étudiant ou Entreprise)
- Validation des données (email, mot de passe)
- Création automatique du profil correspondant

- Attribution du rôle

Connexion :

- Authentification par email/mot de passe
- Gestion de la session utilisateur
- Redirection selon le rôle

Sécurité :

- Mots de passe hachés (ASP.NET Core Identity)
- Protection CSRF sur tous les formulaires
- Validation côté serveur

3.4.2. Module Gestion des profils

Profil Étudiant :

- Informations : Nom, Prénom, Email, Téléphone, Filière, Niveau
- Modification des informations personnelles
- Lien avec le compte Identity via Email

Profil Entreprise :

- Informations : Nom, Secteur, Adresse, Téléphone, Email contact
- Modification des informations
- Lien avec le compte Identity via EmailContact

3.4.3. Module Offres de stage

Fonctionnalités Entreprise :

- Créer une offre (Titre, Description, Durée, Date de début souhaitée)
- Modifier ses offres
- Supprimer ses offres
- Consulter les candidatures reçues

Fonctionnalités Étudiant :

- Consulter toutes les offres disponibles
- Voir les détails d'une offre (entreprise, description, durée)
- Rechercher et filtrer les offres

Informations affichées :

- Titre et description du poste
- Entreprise (nom, secteur)
- Durée en mois
- Date de publication

3.4.4. Module Candidatures

Soumission de candidature (Étudiant) :

- Sélection d'une offre
- Upload du CV en format PDF (max 10 Mo)
- Validation et stockage sécurisé
- Statut initial : "En attente"

Gestion des candidatures (Entreprise) :

- Consultation des candidatures reçues
- Téléchargement des CV
- Acceptation ou refus avec mise à jour du statut
- Notifications automatiques

Suivi des candidatures (Étudiant) :

- Consultation de ses candidatures
- Visualisation du statut (En attente, Acceptée, Refusée)
- Informations sur l'offre et l'entreprise

Workflow :

1. Étudiant postule → Statut "En attente"
2. Entreprise examine → Accepte ou Refuse
3. Statut mis à jour → Notification à l'étudiant

3.4.5. Module Conventions

Création de convention (Admin) :

- Sélection d'une candidature acceptée
- Saisie des dates (signature, début, fin)
- Upload du document PDF signé
- Calcul automatique du statut selon les dates

Statuts automatiques :

- **À venir** : Date actuelle < Date début
- **En cours** : Date actuelle entre Date début et Date fin
- **Terminé** : Date actuelle > Date fin

Consultation :

- Étudiant : Voir ses conventions
- Entreprise : Voir les conventions liées à ses offres
- Admin : Accès complet

Gestion des documents :

- Stockage dans /wwwroot/uploads/conventions/
- Nommage unique avec GUID
- Téléchargement du PDF signé

3.4.6. Module Rapports de stage

Dépôt de rapport (Étudiant) :

- Sélection de sa convention terminée
- Saisie du titre du rapport
- Upload du fichier PDF (max 10 Mo)
- Date de dépôt automatique

Consultation (Admin) :

- Voir tous les rapports déposés
- Télécharger les fichiers PDF
- Informations : Étudiant, Entreprise, Convention liée

Stockage sécurisé :

- Fichiers dans /wwwroot/uploads/rapports/
- Validation du format (PDF uniquement)
- Suppression automatique si rapport supprimé

3.4.7. Fonctionnalités avancées

Recherche et filtrage :

- Recherche d'offres par mots-clés

- Filtrage par secteur, durée, entreprise
- Tri des résultats

Notifications par email :

- Nouvelle candidature → Entreprise
- Candidature acceptée/refusée → Étudiant
- Convention créée → Étudiant et Entreprise
- Rapport déposé → Admin

Tableaux de bord :

- Statistiques pour l'admin (nombre d'offres, candidatures, conventions)
- Graphiques de suivi
- Vue d'ensemble du système

Export de données :

- Export Excel des candidatures
- Export CSV des rapports
- Génération de rapports statistiques

3.5. Gestion de la sécurité

3.5.1. Authentification

- Système Identity avec validation d'email
- Mots de passe hachés (PBKDF2)
- Session sécurisée avec cookies

3.5.2. Autorisation

- Décorateur [Authorize] sur tous les contrôleurs
- Filtrage des données selon le rôle
- Protection des routes sensibles

3.5.3. Upload de fichiers

Validation :

- Type de fichier : PDF uniquement
- Taille maximale : 10 Mo
- Extension vérifiée côté serveur

Stockage :

- Noms de fichiers uniques (GUID)
- Organisation par type (cv/, rapports/, conventions/)
- Suppression automatique lors de la suppression de l'entité

Sécurité :

- Pas d'exécution de fichiers uploadés
- Validation stricte des extensions
- Isolation des fichiers dans wwwroot/uploads/

4. TESTS ET DIFFICULTÉS

4.1. Tests réalisés

4.1.1. Tests fonctionnels

Des tests fonctionnels ont été réalisés pour valider chaque module de l'application.

Test du module Authentification

- Inscription avec différents types de comptes (Étudiant, Entreprise)
- Connexion avec identifiants valides et invalides
- Déconnexion et gestion de session
- Redirection automatique selon le rôle

Test du module Offres de stage

- Création d'offres par l'entreprise
- Modification et suppression d'offres
- Consultation des offres par les étudiants
- Recherche et filtrage des offres

Test du module Candidatures

- Soumission de candidature avec upload de CV
- Consultation des candidatures (Étudiant et Entreprise)
- Acceptation et refus de candidatures
- Mise à jour automatique des statuts

Test du module Conventions

- Création de convention pour candidature acceptée
- Upload de document PDF signé
- Calcul automatique du statut (À venir, En cours, Terminé)
- Consultation selon les rôles

Test du module Rapports

- Dépôt de rapport avec upload PDF
- Téléchargement des rapports
- Lien avec la convention correspondante

4.1.2. Scénarios de test

Scénario 1 : Workflow complet étudiant

1. Inscription en tant qu'étudiant
2. Complétion du profil
3. Consultation des offres disponibles
4. Candidature à une offre avec CV
5. Suivi du statut de la candidature
6. Consultation de la convention après acceptation
7. Dépôt du rapport de stage

Résultat : Workflow complet fonctionnel

Scénario 2 : Workflow complet entreprise

1. Inscription en tant qu'entreprise
2. Complétion du profil
3. Création d'une offre de stage
4. Réception de candidatures
5. Téléchargement des CV
6. Acceptation d'une candidature
7. Consultation de la convention créée

Résultat : Workflow complet fonctionnel

Scénario 3 : Gestion admin

1. Connexion en tant qu'administrateur
2. Consultation de toutes les candidatures acceptées
3. Création d'une convention avec dates
4. Upload du document PDF signé
5. Vérification du statut automatique
6. Consultation des rapports déposés

Résultat : Fonctionnalités admin opérationnelles

4.1.3. Tests de sécurité

Test d'autorisation

- Accès aux pages sans authentification : Bloqué (redirection vers login)
- Accès étudiant aux pages entreprise : Bloqué (erreur 403)
- Accès entreprise aux données d'autres entreprises : Bloqué
- Modification de données par un rôle non autorisé : Bloqué

Test d'upload de fichiers

- Upload de fichier non-PDF : Rejeté avec message d'erreur
- Upload de fichier > 10 Mo : Rejeté avec message d'erreur
- Upload de fichier PDF valide : Accepté et stocké

Test de validation des données

- Soumission de formulaire avec champs vides : Erreurs de validation affichées
- Email invalide lors de l'inscription : Rejeté
- Dates incohérentes (DateFin < DateDebut) : Rejetées

4.2. Difficultés rencontrées et solutions

4.2.1. Problème : NullReferenceException avec les relations

Description du problème Lors de l'affichage des rapports de stage, l'application générait des erreurs NullReferenceException quand on tentait d'accéder aux informations de l'étudiant ou de l'entreprise via les relations.

Cause Entity Framework Core ne charge pas automatiquement les entités liées (Lazy Loading désactivé par défaut). Les propriétés de navigation (Convention, Candidature, Etudiant, etc.) restaient null.

Solution appliquée Utilisation de .Include() et .ThenInclude() pour charger explicitement toutes les relations nécessaires :

```
var rapports = await _context.RapportsStages
    .Include(r => r.Convention)
        .ThenInclude(c => c.Candidature)
            .ThenInclude(cand => cand.Etudiant)
    .Include(r => r.Convention)
        .ThenInclude(c => c.Candidature)
            .ThenInclude(cand => cand.OffreStage)
                .ThenInclude(o => o.Entreprise)
    .ToListAsync();
```

Résultat : Chaque utilisateur accède uniquement aux données pertinentes pour son rôle.

4.2.2. Problème : Autorisation granulaire par rôle

Description du problème Nécessité de filtrer les données affichées selon le rôle de l'utilisateur connecté. Par exemple, un étudiant ne doit voir que ses propres candidatures, une entreprise que les candidatures liées à ses offres.

Solution appliquée Implémentation d'un filtrage contextuel dans chaque contrôleur :

```
if (User.IsInRole("Etudiant")) {
    var etudiant = await _context.Etudiants
        .FirstOrDefaultAsync(e => e.Email ==
User.Identity.Name);
    candidaturesQuery = candidaturesQuery
        .Where(c => c.EtudiantId == etudiant.Id);
}
```

Résultat : Chaque user accède uniquement aux données pertinentes pour son rôle.

4.2.3. Problème : Upload et stockage de fichiers

Description du problème Gestion sécurisée de l'upload de fichiers (CV, conventions, rapports) avec validation du type et de la taille, stockage organisé et suppression automatique.

Solution appliquée

1. Validation stricte : vérification de l'extension (.pdf) et de la taille (max 10 Mo)
2. Nommage unique avec GUID pour éviter les conflits
3. Organisation par dossiers : /uploads/cv/, /uploads/rapports/, /uploads/conventions/
4. Suppression automatique du fichier lors de la suppression de l'entité

```
var uniqueFileName = $"{Guid.NewGuid()}_{file.FileName}";
var filePath = Path.Combine(_environment.WebRootPath,
"uploads/rapports", uniqueFileName);
await file.CopyToAsync(new FileStream(filePath,
FileMode.Create));
```

Résultat : Stockage sécurisé et organisé des fichiers uploadés.

4.2.4. Problème : Calcul automatique du statut des stages

Description du problème Le statut d'un stage (À venir, En cours, Terminé) doit être calculé automatiquement en fonction des dates de la convention et de la date actuelle.

Solution appliquée Calcul dynamique dans le contrôleur lors de l'affichage :

```
var today = DateTime.Now.Date;
string statutStage;

if (today < convention.DateDebut)
    statutStage = "À venir";
else if (today >= convention.DateDebut && today <=
convention.DateFin)
    statutStage = "En cours";
else
    statutStage = "Terminé";
```

Résultat : Statut toujours à jour sans intervention manuelle.

4.2.5. Problème : Lien entre Identity et tables métier

Description du problème Nécessité de lier les comptes Identity (authentification) avec les tables Etudiant et Entreprise (données métier) via l'email.

Solution appliquée

1. Lors de l'inscription, création automatique de l'entité correspondante :
    - Compte Étudiant → Création d'un enregistrement Etudiant avec le même email
    - Compte Entreprise → Création d'un enregistrement Entreprise avec le même email
2. Récupération de l'entité métier via l'email de l'utilisateur connecté :

```
var userEmail = User.Identity.Name;
var etudiant = await _context.Etudiants
    .FirstOrDefaultAsync(e => e.Email == userEmail);
```

Résultat : Liaison automatique et transparente entre authentification et données métier.

5. CONCLUSION

Ce projet de développement d'une application web de gestion des stages a permis de mettre en pratique les connaissances acquises en développement web avec ASP.NET Core MVC et de découvrir les défis concrets liés à la création d'une application complète et fonctionnelle.

Bilan du travail réalisé

L'application développée répond aux objectifs fixés en offrant une solution complète pour la gestion du processus de stage. Le système permet désormais de digitaliser l'ensemble du workflow, depuis la publication des offres jusqu'au dépôt des rapports de stage, en passant par la gestion des candidatures et des conventions.

Les fonctionnalités principales implémentées incluent :

- Un système d'authentification multi-rôles sécurisé avec ASP.NET Core Identity
- La gestion complète des offres de stage par les entreprises
- Un processus de candidature avec upload de CV pour les étudiants
- La gestion des conventions de stage par l'administration avec calcul automatique du statut
- Le dépôt et la consultation des rapports de stage
- Des fonctionnalités avancées de recherche, filtrage, notifications et export de données

L'architecture MVC adoptée et l'utilisation d'Entity Framework Core avec l'approche Code First ont permis de développer une application structurée, maintenable et évolutive. La séparation des responsabilités entre les modèles, les vues et les contrôleurs facilite la compréhension du code et les futures modifications.

Compétences acquises

Ce projet a permis de développer plusieurs compétences techniques et méthodologiques :

Compétences techniques :

- Maîtrise du framework ASP.NET Core MVC et de son architecture
- Utilisation avancée d'Entity Framework Core pour la gestion des données
- Implémentation d'un système d'authentification et d'autorisation avec Identity
- Manipulation de LINQ pour les requêtes de base de données
- Gestion sécurisée de l'upload de fichiers

- Intégration de Bootstrap 5 pour une interface responsive

Compétences méthodologiques :

- Analyse des besoins et conception avec UML
- Approche Code First pour la modélisation de la base de données
- Implémentation d'une architecture RBAC (Role-Based Access Control)
- Résolution de problèmes techniques (NullReferenceException, autorisation granulaire)
- Tests fonctionnels et validation de scénarios complets

Compétences transversales :

- Gestion de projet et respect des deadlines
- Documentation technique et rédaction de rapport
- Autonomie dans la recherche de solutions
- Capacité d'adaptation face aux difficultés techniques

Perspectives et améliorations futures

Bien que l'application soit fonctionnelle, plusieurs améliorations peuvent être envisagées pour enrichir le système :

Améliorations de l'interface utilisateur :

- Ajout de graphiques et statistiques interactifs pour les tableaux de bord
- Amélioration de l'ergonomie avec des animations et transitions
- Mise en place d'un système de thèmes personnalisables

Fonctionnalités supplémentaires :

- Système de messagerie intégrée entre étudiants et entreprises
- Génération automatique de documents PDF (conventions, attestations)
- Calendrier pour visualiser les périodes de stage
- Système d'évaluation des stages par les étudiants et les entreprises

Optimisations techniques :

- Mise en cache pour améliorer les performances
- Pagination avancée pour les grandes listes
- API REST pour permettre l'intégration avec d'autres systèmes

- Application mobile (iOS/Android) avec Xamarin ou React Native

Sécurité renforcée :

- Authentification à deux facteurs (2FA)
- Journalisation des actions sensibles
- Chiffrement des fichiers uploadés
- Politiques de mots de passe renforcées