Projet intégrateur 420-DA3-AS

Phase 2: Réalisation de l'application

Rappel

Le projet est divisé en 3 phases:

- Formation de l'équipe (semaines 1 à 3)
- Analyse: analyse du domaine, des cas d'utilisation et rapport d'analyse (semaines 4 à 7)
  - Analyse du domaine (semaine 4)
  - Analyse des cas d'utilisation (semaines 5-6)
  - Rapport d'analyse (semaine 7)
- Réalisation: production du code, revue et tests de qualité (semaines 8 à 15)
  - Réalisation : classes du domaine (entités) (semaine 8)
  - Réalisation : classe de contexte et configuration de base des entités avec *ModelBuilder* (semaine 9)
  - Réalisation : configuration des associations entre entités (semaine 10)
  - Réalisation : classes d'accès aux données (semaine 11)
  - Réalisation : classes de présentation (semaine 12)
  - Réalisation : classes métier (semaine 13)
  - Réalisation : intégration et qualité du code (semaines 14-15)

Table des matières

Consignes

Vous avez jusqu'au **15 décembre 2024 à minuit** pour compléter la phase de réalisation (production de l'application). Lisez attentivement la section Requis académiques du présent document ***pour voir la répartition des points de l'évaluation***; à partir du cahier de charges de la phase d'analyse, de votre propre analyse des cas d'utilisation ainsi que des documents de conception fournis par l'enseignant, vous devrez réaliser une application qui respecte les fonctionnalités requises – voir la section Fonctionnalités requises de l'application.

Notez que vous ne commencerez pas un projet de programmation vide : les coordonnateurs d'équipe obtiendront une copie d'un projet partiellement créé avec des éléments fournis par l'enseignant. Votre tâche sera de compléter celui-ci en ajoutant votre code et en s'assurant que toutes les parties communes du travail soient fonctionnelles.

Un dossier de conception étoffé avec des diagrammes suggérant une approche pour la réalisation sera inclus dans ce projet. ***Vous n'êtes pas obligés de suivre ces documents de conception; cependant je vous le conseille fortement***. ***Notez aussi que le dossier de conception fournir ne sera pas nécessairement suffisant ou complet pour la réalisation de toutes les fonctionnalités. A vous d'ajouter à votre application ce dont vous avez besoin pour réaliser les fonctionnalités requises***.

Requis académiques

Éléments requis absolument (échec si non-respectés)

- Vous devez **obligatoirement** :
  - **réaliser le projet sous la forme d'une application native (pas web);**
  - **réaliser le projet dans le langage C# (version 10 ou inférieure; la version par défaut de *.NET 6.0 du projet fourni*);**
  - **réaliser le projet pour la plateforme *.NET 6.0;***
  - **utiliser les technologies *Windows Forms*, *Entity Framework Core* ainsi qu'un serveur *Sql Server 2022*;**
  - **implémenter le modèle à trois tiers (au moins minimalement);**
  - **utiliser le système de migrations de *Entity Framework Core;***
  - **utiliser le système de gestion de versions Git pour travailler en commun sur un seul projet; vous NE POUVEZ PAS simplement faire votre partie de votre bord et essayer de rassembler les fichiers de code disparates par la suite : vous DEVEZ utiliser Git; et**
  - **pousser régulièrement vous changements vers github.**
- Il est fortement suggéré de tirer (pull) quand vous commencez à travailler sur le projet (chaque jour que vous travaillez dessus). ***Contactez-moi en cas de problème avec Git***.
- Tout cas catastrophique de manquements de fonctionnalités requises / de travail non complété de votre partie entraînera bien entendu de sévères conséquences. ***N'attendez pas à la fin de la session pour travailler sur le projet!!!!***

Application et code fonctionnel (45%)

- Le code est libre d'erreurs majeures, s'exécute correctement et les fonctionnalités requises sont fonctionnelles :
  - Fonctionnalités implémentées tel que requis (5%)
  - Entités, configuration des entités, vérifications anti-concurrence et données de test inclues (15%)
  - DAOs et instructions EF Core (13%)
  - Services et façades (12%)
- Toute fonctionnalité supplémentaire (en plus des requis) non complétée ou non fonctionnelle doit être gérée avec grâce (correction négative).

Qualité du code (40%)

- Le code est sécuritaire :
  - Validation des données entrées (5%)
  - Visibilité des objets/membres (sécurité interne) (2.5%)
  - Membres et variables fortement typées (5%)
  - Gestion des exceptions (5%)
- Le code est adaptable et professionnel :
  - Méthodes fonctionnellement atomiques (5%)
  - Séparation correcte des éléments fonctionnels entre les trois tiers (5%)
  - Documentation du code (7.5%)
  - Respect des conventions de nommage (5%)

Design de l'application (15%)

- L'interface graphique est professionnelle :
  - Choix de contrôles appropriés, UX claire, facile et intuitive (12%)
  - Interface dynamique : s'adapte en taille (3%)

Données initiales minimales à faire insérer par migration

Votre application doit minimalement insérer automatiquement les données initiales suivantes lors de la création de la base de données par migration (vous pouvez en insérer plus si vous voulez) :

- 3 Rôles, un pour chaque type d'emploi
- 3 Utilisateurs, dont :
  - au moins un ayant le rôle administrateur
  - au moins un ayant le rôle employé d'entrepôt
  - au moins un ayant le rôle employé de bureau
- 1 Entrepôt
- 2 Clients
- 3 Produits (au moins un par client)
- 2 Fournisseurs
- Autant d'adresses que nécessaires

Fonctionnalités requises de l'application

NOTE : les fonctionnalités décrites ci-dessous vont possiblement vous demander d'ajouter des éléments et structures qui ne sont pas spécifiés dans les documents de conception fournis. Vous aurez très probablement besoin de plus de méthodes dans vos services et dans vos DAOs que ce qui est spécifié dans ces documents, voir même une ou plusieurs « stacks » complètes (comme pour les associations produit-ordre de restockage). À vous de voir selon la logique de l'application ce qui est nécessaire pour implémenter ces fonctionnalités. CONTACTEZ-MOI SI VOUS AVEZ BESOIN D'AIDE.

Fonctionnement général :

- Les noms d'utilisateur des utilisateurs doivent être uniques.
- Les mots de passe ne doivent pas être enregistrés en clair (non-encryptés) dans la base de données (utiliser la méthode HashPassword de la classe PasswordService pour les encrypter et ValidatePassword pour valider un mot de passe en clair contre le hash d'un mot de passe enregistré).
- La création d'ordre de restockage devrait être semi-automatisée (créées automatiquement lors de la création d'un ordre d'expédition qui fait tomber la quantité en stock d'un produit à moins de 50% de la quantité en stock visée OU créées manuellement par un administrateur)
- La complétion d'un ordre de restockage (conceptuellement, la réception de celle-ci par un employé d'entrepôt dans l'entrepôt de destination) doit augmenter la quantité en stock disponible du produit par la quantité spécifiée dans l'ordre de restockage.
- La création d'expéditions (Shipment) devrait être semi-automatisée : elle survient conceptuellement lorsqu'un employé d'entrepôt a fini d'emballer les produits d'un ordre d'expédition. Il crée alors une expédition en sélectionnant le service de livraison voulu et une expédition pour ce service de livraison est alors créé (avec numéro de suivi généré automatiquement – utiliser la classe TrackingNumberFactory pour cela) et celui-ci est associé à l'ordre d'expédition.
- La création d'ordres d'expédition (par employés de bureau ou administrateurs) se fait en deux étapes :
  - Création de l'ordre d'expédition initial (statut New).
    - Note, on devrait pouvoir créer une adresse de destination finale directement depuis l'interface de création de l'ordre d'expédition. Un bouton qui déclenche ce processus de création d'adresse peut faire le travail.

  - Ajout de produits et de leurs quantités à l'ordre d'expédition.
    - Note : on ne doit pas pouvoir ajouter une quantité d'un produit supérieure à sa quantité en stock (voir le contrôle windows forms NumericUpDown et ses propriétés Min et Max).
  - Une fois tous les produits ajoutés, création de toutes les associations produit-ordre d'expédition, mise à jour de l'ordre d'expédition (statut Unassigned) et mise à jour des produits pour diminuer leur quantité en stock disponible.
- La modification d'un ordre d'expédition ne peut se faire que si l'ordre d'expédition à le statut New ou Unassigned.
  - La modification des produits associés à une commande ou de leur quantité doit aussi modifier la quantité en stock des produits associés. ***ATTENTION : ceci est compliqué à implémenter : avant d'ouvrir une fenêtre permettant la modification de l'association produit-ordre d'expédition, il faut obtenir et stocker quelque part la quantité initiale pré-modification afin de pouvoir calculer le différentiel et soit diminuer ou augmenter la quantité de produit en stock disponible après la modification. Me contacter pour assistance.***
- on devrait pouvoir créer une adresse pour un entrepôt directement depuis l'interface de création d'entrepôt. Un bouton qui déclenche ce processus de création d'adresse peut faire le travail.

Capacités des utilisateurs de l'application :

NOTE : dans la section suivante, les termes « sélection préalable par recherche filtrée » implique l'utilisation d'un champ de recherche et d'une liste de résultats respectant le critère de recherche pour permettre la sélection d'une entité.

Tous les utilisateurs doivent pouvoir :

- S'authentifier et se connecter à l'application.
- Choisir un de leurs rôles s'ils en possèdent plusieurs

Un **Administrateur** (utilisateur possédant rôle administrateur) doit pouvoir :

- Faire certaines des 4 opérations de base (create, read, update et delete; les CRUD) :
  - Address (Adresse enregistrée) : toutes les CRUD; implique :
    - pour Read, Update et Delete : sélection préalable ***par recherche filtrée*** d'Address existante.
  - Client (clients) : toutes les CRUD; implique :
    - pour Read, Update et Delete : sélection préalable ***par recherche filtrée*** de Client existant

        - pour Create et Update : sélection de Warehouse et d'Address existantes ou, ***optionnellement***, création intégrée de Warehouse et d'Address.
    - Warehouse (Entrepôt) : toutes les CRUD; implique :
        - pour Read, Update et Delete : sélection préalable ***par recherche filtrée*** de Warehouse existante.
        - pour Create et Update : sélection d'Address existante ou, ***optionnellement***, création intégrée d'Address.
    - Product (Produit) : toutes les CRUD; implique :
        - pour Read, Update et Delete : sélection préalable ***par recherche filtrée*** de Product existant
        - Contrainte pour Delete : le produit doit avoir une quantité en stock égale à 0.
        - Pour Create et Update : sélection de Client et de Supplier ou, ***optionnellement***, création intégrée de Client et de Supplier
    - Supplier (Fournisseur) : toutes les CRUD; implique :
        - pour Read, Update et Delete : sélection préalable ***par recherche filtrée*** de Supplier existant
    - ShippingOrder (Ordre d'expédition) : toutes les CRUD; implique :
        - pour Read, Update et Delete : sélection préalable ***par recherche filtrée*** de ShippingOrder existante,
        - Contrainte pour Update : autorisé seulement si statut = 'new' ou 'unassigned'.
        - pour Create et Update : sélection ***par recherche filtrée*** de Product existants à ajouter/retirer à la ShippingOrder, incluant une quantité à ajouter/retirer (ne doit pas être plus que la quantité du produit en stock. Pour la modification de la quantité : la variation de la quantité ne peut pas rendre la quantité de produit en stock inférieure à zéro. Si la variation est négative (retire des produits), la quantité retirée doit être ré-ajouté aux stocks du produit).
        - Pour Create : sélection de Client existant ou, ***optionnellement***, création intégrée de Client et création intégrée d'Address de destination finale.
        - La création finale (avec tous les produits associés) d'une ShippingOrder change la quantité en stock des Product lui étant associée.
    - Shipment (Livraison) : Create et Read seulement (cas spécial : pas Update ou Delete D); implique :
        - pour Read : sélection ***par recherche filtrée*** de Shipment existant
        - pour Create : sélection de ShippingOrder à laquelle associer le shipment (où le ShippingOrder n'a pas déjà de Shipment associée)
    - PurchaseOrder (Ordre de restockage) : Create et Read seulement (cas spécial : pas Update ou Delete D); implique :
        - pour Create : sélection ***par recherche filtrée*** de Product existant.
        - pour Read : sélection ***par recherche filtrée*** de PurchaseOrder existante.

- User (Utilisateur) : toutes les CRUD, implique :
  - pour Create, Read et Delete : sélection ***par recherche filtrée*** d'un User existant.
  - pour Create et Update : sélection de Role(s) a ajouter au User (où le User ne possède pas déjà le Role)
  - pour Create et Update : sélection de Role (où le User possède déjà le Role) à retirer du User
- Role : toutes les CRUD, implique :
  - pour Read, Update et Delete : sélection d'un Role existant.

Un **Employé de bureau** (utilisateur possédant le rôle correspondant) doit pouvoir :

- Procéder à la gestion des ShippingOrder et des Client, et tout ce qui va avec :
  - Création de ShippingOrder; implique :
    - Sélection ***par recherche filtrée*** de Product existants à ajouter/retirer à la ShippingOrder et entrée de quantité pour le produit (ne doit pas être plus que la quantité du produit en stock).
    - Sélection de Client existant.
    - Création intégrée d'une Address de destination finale.
    - La création finale (avec tous les produits associés) d'une ShippingOrder change la quantité en stock des Product lui étant associée.
  - Gestion de Client : Create, Read et Update implique :
    - Création de nouveaux Client et tout ce qui va avec
      - sélection de Warehouse existante.
    - Modification/Visualisation de Client
      - sélection préalable ***par recherche filtrée*** de Client existant
  - Gestion des Product : toutes les CRUD
    - Création de nouveau Product avec sélection ou création de Supplier et ce qui va avec.
    - Pour Read, Update et Delete de Product existant :
      - sélection ***par recherche filtrée*** de Product existant.
      - suppression seulement si quantité en stock du Product est 0.
  - Gestion des Supplier : toutes les CRUD
    - Création de nouveau Supplier.
    - Pour Read, Update et Delete de Supplier existant :
      - sélection ***par recherche filtrée*** de Product existant.

Un **Employé d'entrepôt** (utilisateur possédant le rôle correspondant) doit pouvoir :

- Visualiser sous forme de liste les ShippingOrder de Client assignés au Warehouse où il travaille et dont le statut est 'Unassigned' (non-assignées à un autre User employé d'entrepôt).

- Visualiser sous forme de liste les ShippingOrder assignées à lui-même et dont le statut est 'Processing'.
- Visualiser sous forme de liste les ShippingOrder de Client assignés au Warehouse où il travaille et dont le statut est 'Packaged' (en attente d'être ramassées par un service de livraison).
- Visualiser sous forme de liste les PurchaseOrder non-complétées (dont le statut est 'New') dont la destination est le Warehouse où il travaille.
- S'assigner une ShippingOrder nouvellement créée et non-assignée à un autre User employé d'entrepôt (change aussi automatiquement le statut de la ShippingOrder à 'Processing').
- Compléter le ramassage (Processing) d'une ShippingOrder lui étant assignée dont le statut est 'Processing' (Change le statut de la ShippingOrder à 'Packaged' et crée automatiquement un Shipment pour la ShippingOrder).
  - Ceci inclus de sélectionner le fournisseur pour la création du Shipment.
- Marquer une ShippingOrder d'un Client assignés au Warehouse où il travaille dont le statut est 'Packaged' comme 'Shipped'.
- Marquée une PurchaseOrder non-complétée (dont le statut est 'New') dont la destination est le Warehouse où il travaille comme ayant été reçue (change le statut de la PurchaseOrder à 'Completed' et change la quantité en stock du Product de la PurchaseOrder).

Le **Système** (actions automatiques) doit pouvoir :

- Lors de la création d'une ShippingOrder, pour chaque Product associé à cette ShippingOrder dont la quantité en stock tombe en bas de 50% de sa quantité en stock visée, création automatique d'une PurchaseOrder pour le Product avec une quantité de restockage prévue pour ré-atteindre la quantité en stock visée.

Grille de correction

Élément de compétence : Analyser le projet de développement de l'application. (00SS.1)	
Critères de performance	**Poids**
1.2 Détermination correcte des tâches à effectuer. (app fonctionnelle – fonctionnalités)	/5
Élément de compétence : Préparer l'environnement de développement informatique. (00SS.2)	
Critères de performance	**Poids**
2.1 Installation correcte des logiciels et des bibliothèques sur la plateforme hôte.	**absolu**
Élément de compétence : Préparer la ou les bases de données. (00SS.3)	
Critères de performance	**Poids**
3.1 Création ou adaptation correctes de la base de données locale ou de la base de données distante. (app fonctionnelle – Entités configuration fonctionnelle pour migrations)	/3
3.2 Insertion correcte des données initiales ou des données de tests. (app fonctionnelle – Entités données initiales)	/5
3.3 Respect du modèle de données. (app fonctionnelle – Entités configuration)	/5
Élément de compétence : Générer ou programmer l'interface graphique. (00SS.4)	
Critères de performance	**Poids**
4.1 Choix et utilisation appropriés des éléments graphiques pour l'affichage et la saisie. (UI-UX)	/12
4.3 Adaptation de l'interface en fonction du format d'affichage et de la résolution. (UI-UX)	/3
Élément de compétence : Programmer la logique applicative. (00SS.5)	
Critères de performance	**Poids**
5.1 Programmation ou intégration correctes de mécanismes d'authentification et d'autorisation. (app fonctionnelle – Services services limités par façade)	/2
5.2 Programmation correcte des interactions entre l'interface graphique et l'utilisatrice ou l'utilisateur. (app fonctionnelle – Services fonctionnalités fonctionelles)	/10
5.3 Choix approprié des clauses, des opérateurs, des commandes ou des paramètres dans les requêtes à la base de données. (app fonctionnelle – DAOs requetes)	/10
5.4 Manipulation correcte des données de la base de données. (app fonctionnelle – DAOs actions)	/3
5.5 Programmation correcte de la synchronisation des données. (app fonctionnelle – Entités anti-concurrence)	/2
5.8 Application rigoureuse des techniques de programmation sécurisée. (qualité du code – validation, visibilité, typage)	/12.5

Élément de compétence : Contrôler la qualité de l'application. (00SS.6)	
Critères de performance	**Poids**
6.2 Revues de code et de sécurité rigoureuses. (fonctions atomiques, exceptions)	/10
6.4 Respect des procédures de suivi des problèmes et de gestion des versions.	**absolu**
6.5 Respect des documents de conception. (qualité du code – conventions et 3 tiers)	/10
Élément de compétence : Rédiger la documentation. (00SS.8)	
Critères de performance	**Poids**
8.1 Détermination correcte de l'information à rédiger. (qualité du code - doc)	/2.5
8.2 Notation claire du travail effectué. (qualité du code - doc)	/5

