E24 – POO III - Travail pratique 3 (22%)

I - Objectif de ce TP

Ce travail pratique (TP) vise à évaluer votre compréhension des notions vues en cours à savoir :

- Création du modèle de classes;
- Création des contrôleurs;
- Création des vues ASP.NET Core;
- Validation de cohérence des données;
- Accès à une base de données avec Entity Framework Core;
- Persistance des données;
- Utilisation de LINQ;
- Respect des bonnes pratiques orientées objet;
- Respect des principes SOLID.

II - Contexte

Ce travail doit être effectué en groupe de deux étudiants. Le même groupe sera maintenu pour le TP4. Vous devez créer un dépôt privé sur GitHub et m'envoyer une invitation en tant que collaborateur.

Mon compte GitHub est : **hinault**.

Vous devez déposer à partir de Lea un fichier contenant les informations suivantes :

- Le rapport PDF;
- Un zip du code de votre application.

III - Date de remise

Votre travail doit être remis au plus tard le mercredi 28 aout à 23h59.

IV - Critères d'évaluation

Votre travail doit respecter l'ensemble des critères suivants :

- Le code doit compiler;
- Le code ne doit pas comporter d'erreurs d'exécution;
- Le code doit réaliser toutes les fonctionnalités demandées;
- Le code doit respecter les bonnes pratiques et les principes SOLID;
- L'application doit être en français;
- Qualité du rapport.

V - Grille de correction

	Excellent	Fonctionnel	Minimal	Insuffisant
Capacité 1 : Programmer les interactions avec une base de données (couche modèle)	**Interaction avec la base de données :** • Modélisation adéquate et optimale de l'application selon le besoin; • Utilisation adéquate en tout temps des types de données ; • Les DataAnnotations sont bien utilisés en tout temps pour répondre aux besoins ; • Implémentation optimale de l'accès aux données • Initialisation optimale des données de tests.	**Interaction avec la base de données :** • Modélisation adéquate et presque optimale de l'application selon le besoin; • Utilisation presque adéquate des types de données ; • Les DataAnnotations sont bien utilisées et répondent presque bien aux besoins; • Initialisation presque optimale des données de tests.	**Interaction avec la base de données :** • Modélisation de l'application répondant presque au besoin; • Utilisation partiellement adéquate des types de données ; • Les DataAnnotations sont la plupart du temps bien utilisés et répondent partiellement aux besoins; • Initialisation partielle des données de tests.	**Interaction avec la base de données** • Modélisation de l'application ne répondant pas au besoin; • Utilisation des types de données rarement adéquats ; • Les DataAnnotations sont rarement utilisés adéquatement; • Les données de tests ne sont pas initialisées adéquatement.
Capacité 2 : Programmer la logique de contrôle des informations (couche contrôleur)	**Mise en place des contrôleurs :** • Programmer une logique applicative fonctionnelle et optimale en respectant l'ensemble des éléments identifiés ; • Appliquer, en tout temps, l'ensemble des bonnes pratiques en programmation (principes SOLID, nomenclature, indentation, documentation,etc.) ; • Valider adéquatement la cohérence des données ; • Appliquer des tests complets permettant de valider que le fonctionnement répond à toutes les exigences; • Le code compile.	**Mise en place des contrôleurs :** • Programmer une logique applicative fonctionnelle et optimale, mais pouvant comporter des lacunes mineures. • Appliquer, en tout temps, l'ensemble des bonnes pratiques en programmation (principes SOLID, nomenclature, indentation, documentation, etc.) avecdes manquements mineurs ; • Valider presque adéquatement la cohérence des données ; • Appliquer des tests complets ou presque permettant de valider que lefonctionnement répond à toutes les exigences; • Le code compile.	**Mise en place des contrôleurs :** • Programmer une logique applicative fonctionnelle, sans être optimale ; • Appliquer, la plupart du temps, l'ensemble des bonnes pratiques en programmation (principes SOLID, nomenclature, indentation, documentation, etc.) ; • Valider partiellement la cohérence des données ; • Appliquer des tests sommaires permettant de valider le fonctionnement minimal de l'application; • Le code ne compile pas.	**Mise en place des contrôleurs :** • Programmer une logique applicative non fonctionnelle ; • Appliquer, rarement, l'ensemble des bonnes pratiques en programmation (principes SOLID, nomenclature, indentation, documentation, etc.) ; • Valider rarement la cohérence des données; • Appliquer des tests ne permettant pas de valider le fonctionnement minimal de l'application; • Le code ne compile pas.
Capacité 3 : Développer des interfaces (couche vue)	**Création des pages :** • Créer une interface graphique fonctionnelle et complète ; • Choisir, positionner et paramétrer les éléments graphiques pertinents et fonctionnels en tout temps ; • Organiser correctement l'ensemble du contenu dans les différentes pages ; • Utiliser adéquatement en tout temps Razor, les Tags Helpers et HTML Helpers.	**Création des pages :** • Créer une interface graphique fonctionnelle,mais pouvant comporter des lacunes mineures ; • Choisir, positionner et paramétrer les éléments graphiques pertinents et fonctionnels, mais pouvant comporter quelques lacunes ; • Organiser correctement l'ensemble contenu dans les différentes pages, mais avec des éléments qui ne sont pas placés de manière optimale ; • Utiliser adéquatement, mais avec des manquements mineurs, Razor, les Tags Helpers et HTML Helpers.	**Création des pages :** • Créer une interface graphique fonctionnelle partiellement; • Choisir, positionner et paramétrer sommairement les éléments graphiques ; • Organiser partiellement son contenu dans les différentes pages ; • Utiliser partiellement Razor, les Tags Helpers et HTML Helpers.	**Création des pages :** • Créer une interface graphique non-fonctionnelle; • Choisir, positionner et paramétrer les éléments graphiques non-pertinents et non-fonctionnels ; • Le contenu n'est pas organisé correctement dans les différentes pages ; • Utiliser rarement Razor, les Tags Helpers et HTML Helpers.

VI - Miniprojet banque Tardi

Votre équipe a été engagée par la banque Tardi pour mettre en place son application de gestion des transactions bancaires. Le projet est développé en plusieurs phases.
La phase 1 qui est réalisée par votre équipe doit permettre de :

- Gérer les clients
    - Affichage de la liste complète des clients
    - Recherche d'un client
    - Ajout/suppression/modification d'un client
- Gérer les comptes bancaires
    - Ajout/modification d'un compte bancaire
    - Ajout d'une opération bancaire
    - Consultation de l'historique des opérations

Gestion des clients

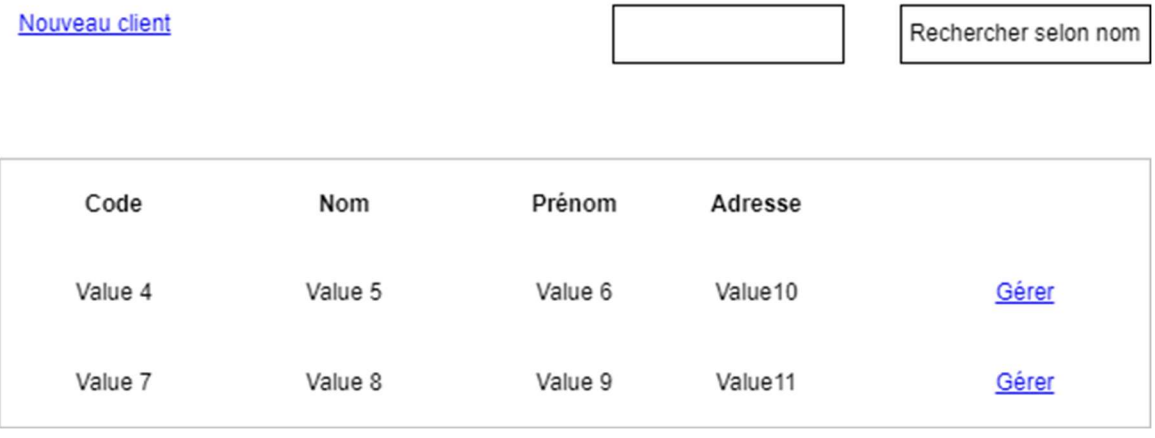
Un client a les caractéristiques suivantes :

- Un **code client** qui est généré automatiquement par la banque et fait office d'identifiant du client. Le code est affiché dans le formulaire de création du client;
- Un **nom**, qui est un champ texte obligatoire et qui ne doit pas dépasser 150 caractères;
- Un **prénom**, qui est un champ texte obligatoire et qui ne doit pas dépasser 150 caractères;
- La **date de naissance**, qui est un champ date et est obligatoire;
- L'**Adresse**, champ texte obligatoire de 250 caractères;
- Le **code postal**, champ texte obligatoire. Doit être un code postal canadien valide;
- **NbDecouverts**, entier initialisé à 0 et dont la valeur n'est pas saisie lors de la création du client;
- **Téléphone** : champ texte. Doit être un numéro de téléphone canadien valide.

Les clients de la banque doivent avoir entre 15 ans et plus. Pour un client qui a moins de 18 ans, on doit obligatoirement renseigner les informations supplémentaires suivantes lors de la création du client :

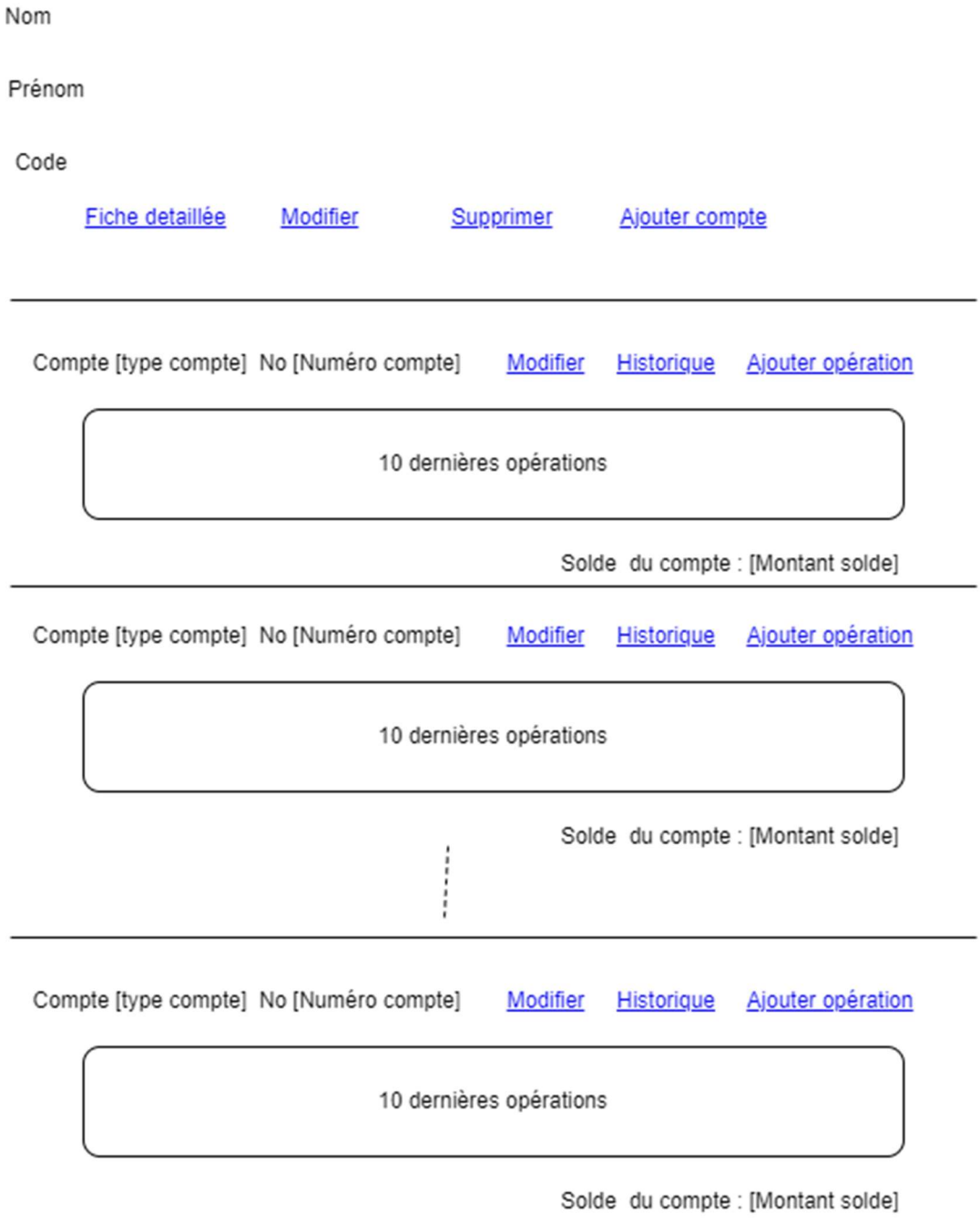
- **Nom Parent** : champ texte de 150 caractères;
- **Téléphone Parent** : champ texte. Doit être un numéro de téléphone canadien valide.

La page d'accueil de l'application doit afficher par défaut la liste des clients (Code, Nom, prénom et Adresse), un bouton **Nouveau** permettant de créer un nouveau client, un champ et un bouton de filtre permettant de rechercher les clients selon leur nom.



Pour chaque client de la liste, un bouton « **Gérer** » doit permettre d'accéder à la page de gestion du client.

La page de gestion du client est l'emplacement de gestion de toutes les opérations bancaires associées au client et devrait ressembler à ce qui suit :



Spécifications des boutons

- **Fiche détaillée** permet d'afficher toutes les informations sur le client;
- **Modifier** permet de modifier les informations sur le client;

- **Supprimer** permet de supprimer directement le client;
- **Ajouter compte** permet d'afficher le formulaire de création d'un compte.
- **Modifier** [dans la section compte] permet de modifier les informations du compte;
- **Historique** permet de consulter la liste de toutes les opérations sur le compte du client;
- **Ajouter une opération** permet d'ajouter une opération au compte du client.

Toutes les actions effectuées doivent retourner sur la page de Gestion du client suite à l'enregistrement avec succès.

Gestion des comptes

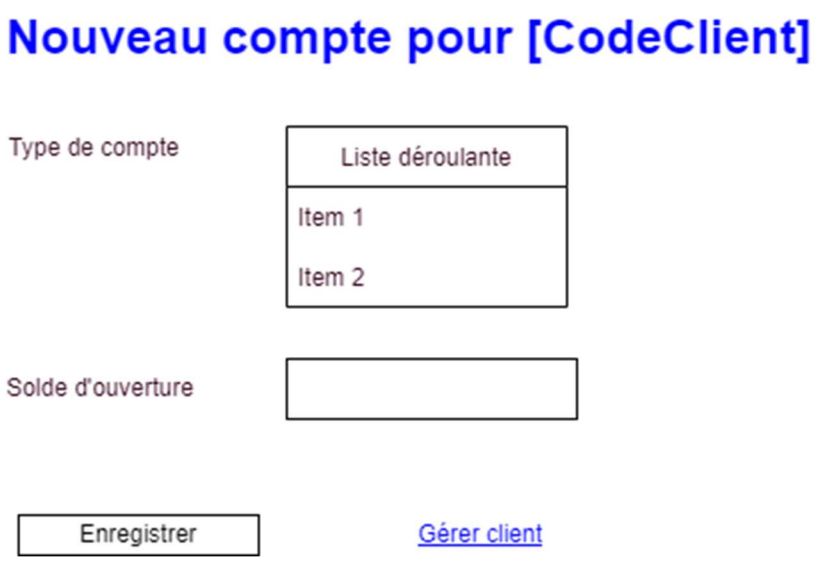
La page de création de compte permet de :

- Sélectionner le type de compte;
- Renseigner le solde d'ouverture du compte.

Ci-dessous les informations sur les différents types de comptes offerts par la banque

Identifiant	Libelle	TauxInteret	TauxInteretDecouvert
10	Chèque	0	7
11	Épargne	2	0
16	Compte de placement garanti	4	0

Le solde d'ouverture doit être un montant positif. Lorsqu'une valeur supérieure à 0 est saisie, automatiquement une opération de crédit avec pour libellé « solde d'ouverture » est ajoutée au nouveau compte.



Les numéros de compte sont générés en respectant le format d'un numéro de compte canadien:

- Un numéro à cinq chiffres qui représente le transit du centre bancaire. Pour la banque Tardi, ce numéro est 00234 et pourrait changer;
- Un numéro à trois chiffres qui représente l'institution bancaire. Pour la banque Tardi, ce numéro est 001 et pourrait changer;
- Le numéro de compte qui est composé de 7 chiffres. Il commence par l'identifiant du type de compte et contient ensuite 5 chiffres auto-incrémenté. Exemple s'il existe deux comptes de types Chèque et deux comptes de type Épargne, ils doivent avoir pour numéro de compte respectivement 1000001 et 1000002 pour les comptes chèques et 1100001 et 1100002 pour les comptes d'épargne.

Modification des comptes

La page de modification du compte permet de modifier le type de compte et le solde d'ouverture uniquement si aucune opération n'a été passée sur le compte.

Gestion des transactions

Le bouton « Ajouter opération » permet d'afficher la page de création des transactions.

La page permet à l'utilisateur de sélectionner le **type d'opération** (Débit ou Crédit), de renseigner le montant de l'opération et le libellé de l'opération.

La **date d'opération** est initialisée par défaut à la date du jour et ne peut pas être modifiée.

Le **montant** de l'opération doit être une valeur positive.

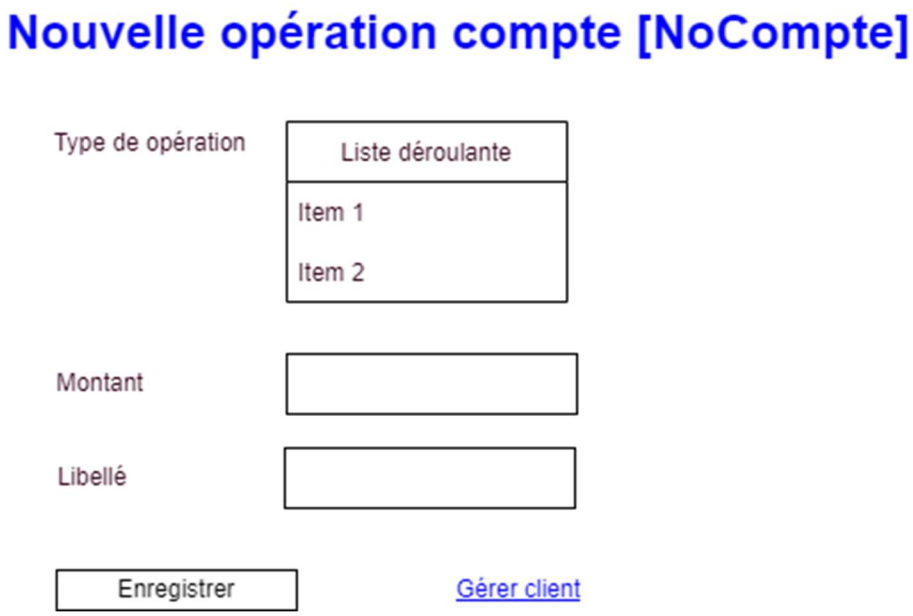
Le **libellé** est un champ obligatoire permettant de donner une description de l'opération.

L'Identifiant de l'opération est un champ auto-incrémenté.

À l'enregistrement, le solde du compte est mis à jour selon le type d'opération.

Seuls les comptes de type chèque autorisent les soldes négatifs. Si le solde du compte devient négatif à la suite d'une opération :

- Ajouter une opération de débit ayant pour libellé « Découvert » de 10$ au compte du client;
- Incrémenter le champ NbDecouverts du client.



Le bouton « **Historique** » affiche la liste complète des opérations sur le compte. On doit être en mesure de filtrer les opérations selon le type. On ne peut pas modifier ou supprimer une opération. Aucun bouton ne doit donc être disponible dans la liste.

VI - Travail à faire

- À partir des informations mises à votre disposition, vous devez modéliser le diagramme de classes du système;
- À partir du diagramme de classes, vous devez créer le modèle de données, puis générer la base de données en utilisant Entity Framework Core. Il est recommandé d'utiliser SQLite comme SGBD;
- Vous devez ensuite implémenter les fonctionnalités de l'application selon les instructions données;
- Vous devez écrire une classe qui permettra d'initialiser la base de données avec des données de tests. Toutes les tables doivent être remplies.

VII – Rapport PDF

Vous devez rédiger un rapport PDF à remettre contenant :

- Le diagramme de classes ou modèle physique de données;
- Un miniguide d'utilisation de l'application avec capture d'écran;
- La liste des difficultés que vous avez rencontrées.