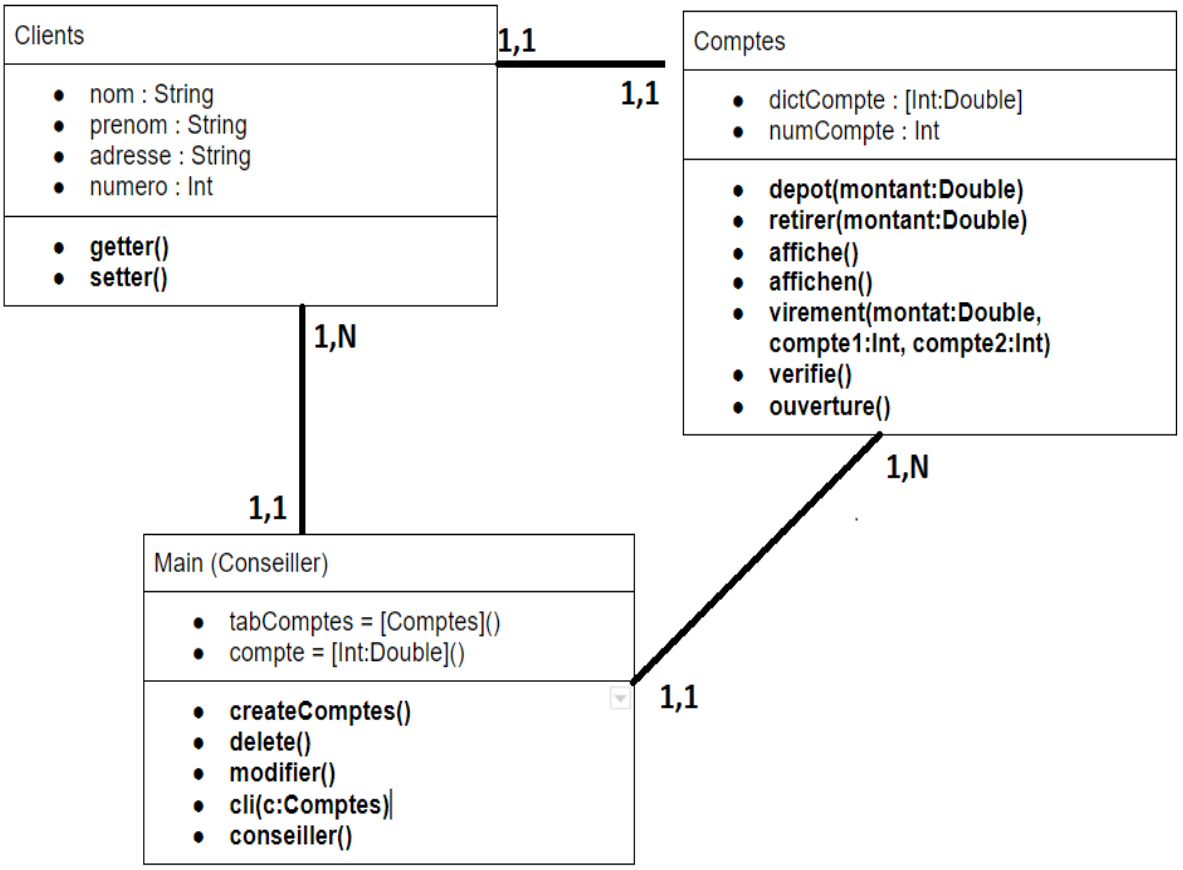
Rapport de projet IOS

DIAGRAMME DE CLASSE :



- Un client peut avoir seulement un seul compte et un seul conseiller.
- Un compte est possédé par un seul client et peut être géré par plusieurs conseillers.
- Un conseiller gère plusieurs clients ainsi que plusieurs comptes clients.

CHOIX FAITS :

Le programme comporte :

- class Clients : Cette classe contient les informations personnelles du clients.
- class Comptes : Cette classe permet à la création d'un compte pour un client avec son numéro de compte qui va être utile pour les virements entre différents clients. Le numéro de compte fonctionne par la taille du tableau de compte + 1, donc le premier inscrit aura le numéro de compte 1.
- main : Le fichier main.swift permet au fonctionnement du programme, mais aussi des manipulations par le conseiller et par le client grâce à des switch. Le programme se lance tout de suite avec le conseiller qui est nous même.

Les fonctionnalités qui doivent être implémentées sont les suivantes :

- Le conseiller peut
    - ouvrir un compte client (**createComptes()**) en s'assurant que le compte n'existe pas.
      Une série de **readLine()** est utile pour entrer les informations nécessaires à la création, puis en l'ajoutant dans la liste des comptes avec la méthode **append**.
    - de clôturer un compte client qui existe (**delete()**). L'utilisation de la méthode **remove** dans une liste suffit pour supprimer totalement le compte de la liste.
    - d'afficher la liste des clients de la banque grâce à un tableau qui contient les comptes clients (**tabComptes = [Compte]()**).
      Une itération en boucle du tableau avec un **print**.
    - de modifier en mettant à jour l'adresse ou le numéro de téléphone du client (**modifier()**).
      Le choix est laissé au conseiller, soit il choisit de modifier l'adresse, soit le numéro de téléphone, puis entre la modification voulu du client.

- Le client peut
    - se connecter à son compte en entrant ses informations personnelles, ce qui le renvoie à la fonction des actions du client (**cli(c:Comptes)**).
    - déposer de l'argent dans son compte courant (**depot(montant:Double)**). L'argent est systématiquement déposer dans le courant et non autre part, le programme doit vérifier tout d'abord qu'il existe bel et bien un compte courant, sinon il le crée.
    - retirer de l'argent (**retirer(montant:Double)**). Même système, l'argent est retiré seulement depuis le compte courant du client.
    - afficher son solde de ses différents comptes (**affiche()**) et son numéro de compte **affichen()** en cas d'oubli
    - effectuer des virements entre ses différents comptes tels que le compte courant, le livret A et l'épargne (**virement(montant:Double, compte1:Int, compte2:Int)**), compte1 est le compte source et le compte2 est la destination.

Pour le client, le programme doit vérifier si le client à au minimum deux comptes pour pouvoir se faire des virements. Sinon, le client doit ouvrir un autre compte que le courant tels que le livret A qui doit avoir au moins 10 euros pour rester ouvert et l'épargne, un minimum de 200 euros. L'argent provient seulement du compte courant, si celui-ci n'a pas assez, le client doit d'abord déposer plus d'argent. C'est alors grâce à la fonction **ouverture()**, on demande au client s'il veut ouvrir un compte livret A si le courant à >= 10E et un compte épargne si il a >= 200E avec la commande "Y" pour oui et "N" pour non. S'il décide d'ouvrir, la somme nécessaire à l'ouverture est retirée au dépôt.

Les virements depuis le compte Livret A et Épargne doivent suivre une règle stricte, un minimum doit rester sur le compte pour qu'elle soit ouverte. Si le client décide de faire un virement depuis le livret A d'un montant qui fait que la somme totale restante du livret A soit inférieure à 10E, alors le virement se fait avec le montant choisi ajouté au reste du compte.

Exemple :

- Livret A = 11 euros, Courant = 0 euro
- Virement de 2 euros vers le compte courant donc : 11 - 2 = 9 euros, 9 < 10.
- Courant = 11 euros

Le compte épargne suit le même principe, mais la somme versée doit être la somme totale du compte épargne lorsque le client décide de faire un virement depuis ce compte.

BONUS :

La fonctionnalité ajoutée est le virement entre les clients qui ont ouvert un compte dépôt. Le client peut alors choisir la somme du virement, et l'envoyer à un autre client de la liste des comptes de la banque grâce à son numéro de compte, et à son nom et prénom.

Il suffit d'itérer sur la liste des comptes et de trouver le numéro de compte **numCompte** ainsi que le nom et prénom, et ces informations doivent correspondre à la saisie de l'utilisateur. Si elle est correcte, on utilise la fonction **retirer(montant:Double)** sur le compte du client voulant verser de l'argent à un autre qui reçoit l'argent grâce à la fonction **depot(montant:Double)**. Il faut aussi prendre en compte le fait que le client n'a pas d'argent dans le compte courant et le fait que le client cible ne possède pas encore de compte courant ouvert.

LES DIFFICULTÉS :

Au départ la création des classes à été nouveau mais cela a été assimilé rapidement. Cependant, le diagramme de classe devait comporter une classe Courant, LivretA et Epargne en plus, cette façon-là n'a pas abouti à ce que je voulais

car je n’arrivais pas à lier la création de compte en même temps que la création des différents comptes.

CONCLUSION :

Le projet est terminé avec toutes les fonctionnalités obligatoires à implémenter, cependant, je pense que mon code à besoin quand même d’être optimisé. L’ajout de nouvelles fonctions devait être plus que seulement une seule.