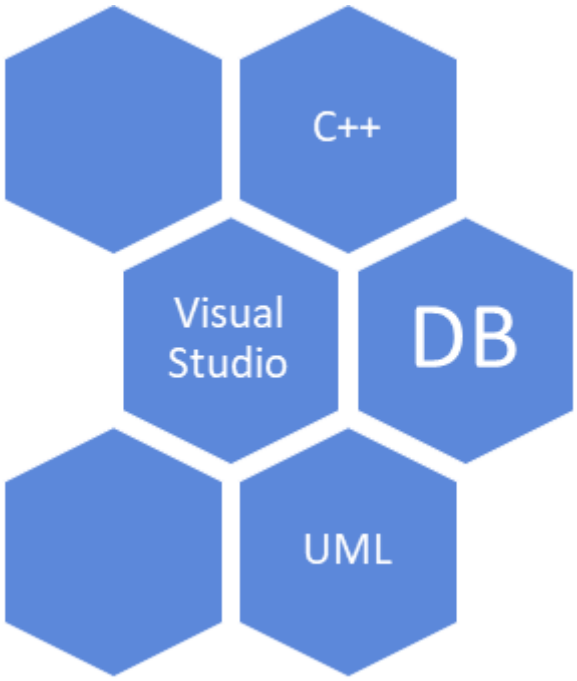
Livrable 2

Programmation Orientée Objet



Groupe 12:

KECILI Hichem

ZAMOUCHE Nadir

GUELLATI Mohamed

HAMHAMI Elrayan

25/11/2021

cesi EXIA ÉCOLE SUPÉRIEURE D'INFORMATIQUE

Sommaire

Introduction :

Dans cette première partie de la réalisation de notre application nous allons tout d'abord commencer par l'étude et la conception de notre application pour cela nous allons utiliser des diagrammes UML, et la mise en place de notre environnement de travail.

Contexte :

Une nouvelle entreprise développe son système d'information. Son cœur d'activité est la vente en ligne de composants électroniques. Nous devons concevoir et réaliser une solution digitalisant certains de ses processus métiers en se basant sur une interview et un cahier des charges fourni par l'entreprise.

I. Modélisation et conception de l'application :

1. Diagramme de cas d'utilisation :

A. Qu'est-ce qu'un diagramme de cas d'utilisation ?

Avant de se lancer dans la réalisation de notre application, Il faut comprendre, clarifier et structurer les attentes et les besoins de notre client, pour cela nous avons utilisé un diagramme de cas d'utilisation.

- Son but :
  - Modéliser les besoins des utilisateurs.
  - Identifier les grandes fonctionnalités et les limites du système.
  - Représenter les interactions entre le système et ses utilisateurs.

- Ces éléments se composent :
  - Des acteurs : Les acteurs se représentent sous la forme d'un petit personnage (stick man) ou sous la forme d'une case rectangulaire (appelé classeur) avec le mot clé « actor ». Un acteur est un utilisateur externe au système. Cela peut être :

- ❖ Une personne.
- ❖ Du matériel (capteurs, moteurs, relais…).
- ❖ Un autre système.

Remarque : Quelque fois, nous utilisons le stick man si l'acteur est humain le classeur si l'acteur est matériel ou un autre système.

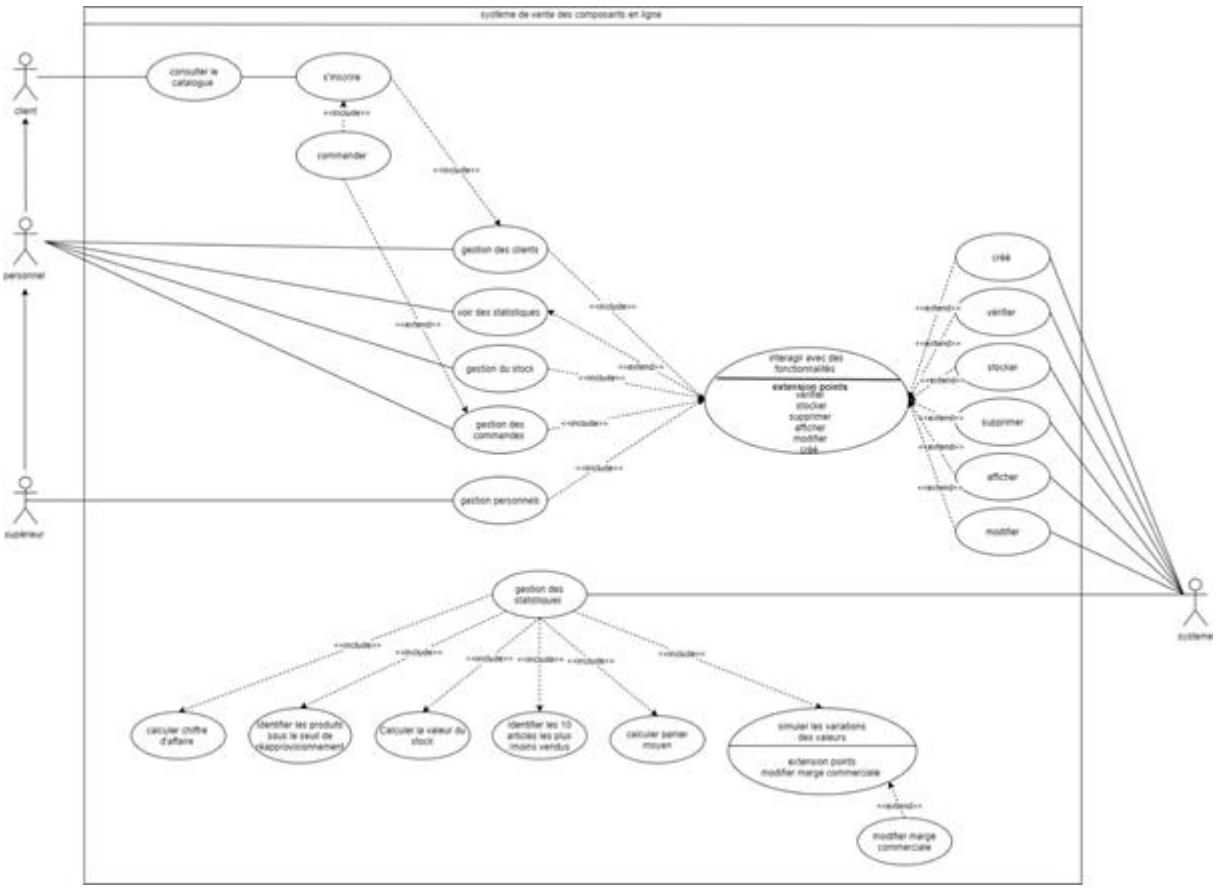
- ➢ Des cas d'utilisation : Le cas d'utilisation représente une fonctionnalité du système (visible de l'extérieur du système). Un cas d'utilisation se représente par une ellipse contenant le nom du cas d'utilisation (phrase commençant par un verbe à l'infinitif) et optionnellement un stéréotype au-dessus du nom. Les différents cas d'utilisation peuvent être représentés à l'intérieur d'un même rectangle représentant les limites du système.

- ➢ Des relations entre les acteurs les cas d'utilisation : A chaque acteur est associé un ou plusieurs cas d'utilisations, la relation d'association peut aussi être appelée relation de communication. Elle est représentée par un trait reliant l'acteur et le cas d'utilisation
- ➢ Des relations entre cas d'utilisation : Tout en faisant attention de ne pas tomber dans le piège d'une décomposition fonctionnelle hiérarchique, nous pouvons compléter le diagramme par d'autres cas d'utilisation (non lié à des acteurs mais à d'autre cas d'utilisation) qui préciseront le diagramme. Parmi eux utilisé dans notre diagramme :

  - ❖ Relation d'inclusion : Dans un diagramme des cas d'utilisation, cette relation est représentée par une flèche pointillée reliant les 2 cas d'utilisation et munie du stéréotype « include ». L'inclusion permet de partager une fonctionnalité commune entre plusieurs cas d'utilisation et de décomposer un cas d'utilisation complexe en décrivant ses sous fonctions.

  - ❖ Relation d'extension : Comme la relation d'inclusion, la relation d'extension enrichit un cas d'utilisation par un autre cas d'utilisation de sous fonction mais celui-ci est optionnel. Cette relation est représentée par une flèche en pointillée reliant les 2 cas d'utilisation et munie du stéréotype « extend ».

B. Le Diagramme :



C. Explication du diagramme :

Notre client va consulter le catalogue puis s'inscrire et avoir l'accès sur son compte pour commander,

Notre acteur personnel va hériter les fonctionnalités du client, il est désigné par l'entreprise pour gérer notre application. Ce personnel va Gérer la gestion des différents processus de l'entreprise (supprimer, créé, afficher, modifier), par exemple la « gestion des clients », « gestion des stocks » … depuis le système. Le supérieur est un acteur qui héritera les fonctionnalités du personnel et du client, il aura aussi la gestion des personnes. Le système permet de vérifier les comptes et paiement … et de géré les commandes des clients. Il permet aussi de stocker les donnes des gestions donne l'accès au personnels et supérieur pour gérer diverses gestions, et aussi faire aussi la gestion des statistiques et faire les calculs et les afficher à l'administrateur.

2. Diagramme de séquence:

A. Qu’est-ce qu’un diagramme de séquences ?

- Un diagramme de séquence est un type de diagramme d'interaction, car il décrit comment et dans quel ordre plusieurs objets fonctionnent ensemble. Ces diagrammes sont utilisés à la fois par les développeurs logiciels et les managers d'entreprises pour analyser les besoins d'un nouveau système ou documenter un processus existant. Les diagrammes de séquence sont parfois appelés diagrammes d'événements ou scénarios d'événements.

Il sert à :

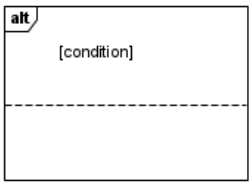
- Représenter les détails d'un cas d'utilisation UML
- Modéliser le déroulement logique d'une procédure, fonction ou opération complexe
- Voir comment les objets et les composants interagissent entre eux pour effectuer un processus.
- Schématiser et comprendre le fonctionnement détaillé d'un scénario existant ou à venir

Ces éléments se composent :

La boite d’activation : Représente le temps nécessaire pour qu'un objet accomplisse une tâche. Plus la tâche nécessite de temps, plus la boîte d'activation est longue.

Acteur : Montre les entités qui interagissent avec le système ou qui sont extérieures à lui.

Symbole de ligne de vie : Représente le passage du temps qui se prolonge vers le bas. Cette ligne verticale en pointillés montre les événements séquentiels affectant un objet au cours du processus schématisé. Les lignes de vie peuvent commencer par une forme rectangulaire avec un intitulé ou par un symbole d'acteur.



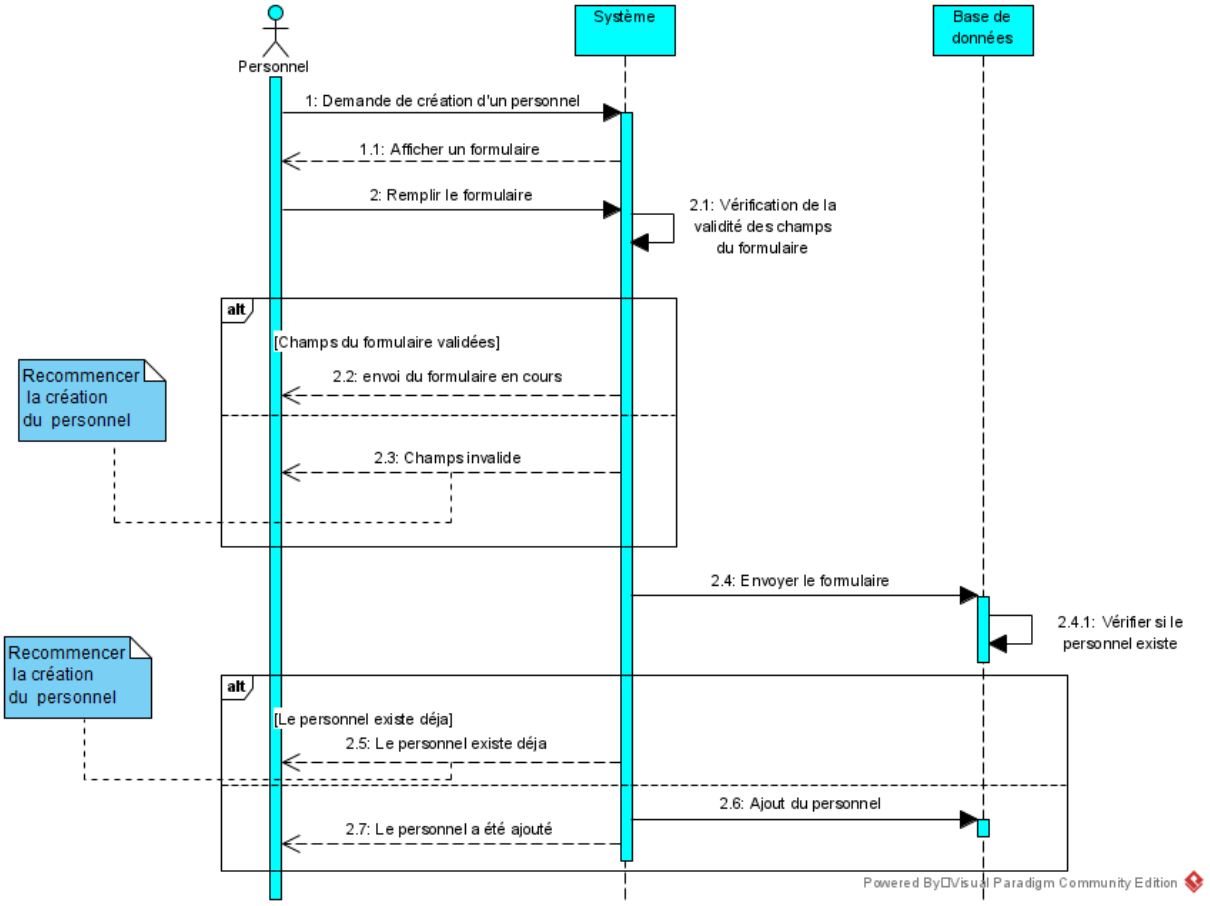
Symbole de fragments multiple alternatifs : On utilise ce symbole pour modéliser des scénarios ou une situation qui peuvent avoir deux comportements possibles, une seule des deux branches est réalisée selon la condition donnée.

Et tant d'autres éléments mais nous ne citerons que les principaux éléments qui ont composé nos diagrammes de séquence.

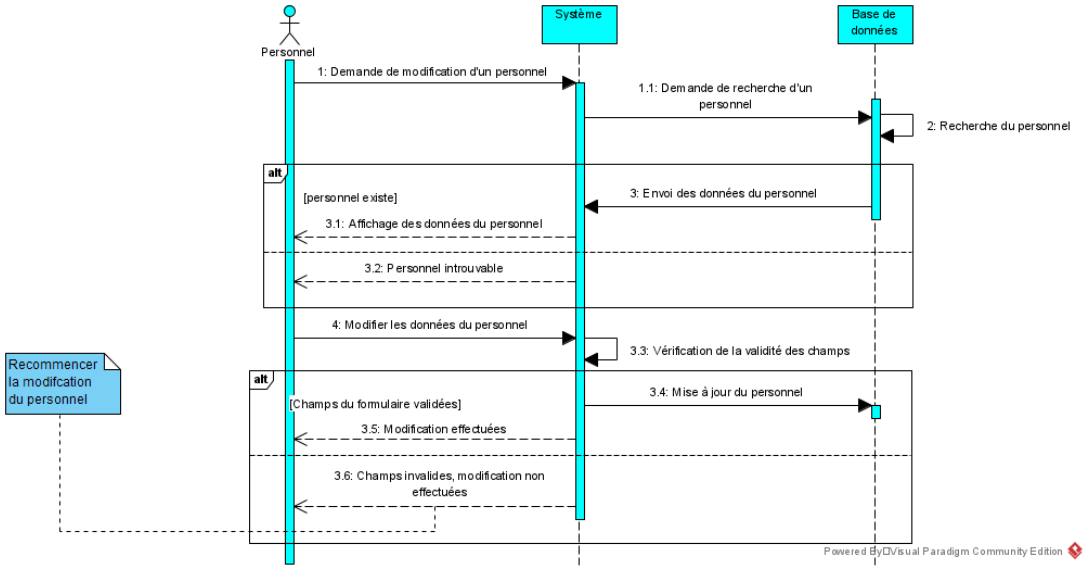
B. Les Diagrammes :

Pour ce diagramme nous en avons réalisé 4 chacun explique une des quatre fonctionnalités de la gestion du personnel :

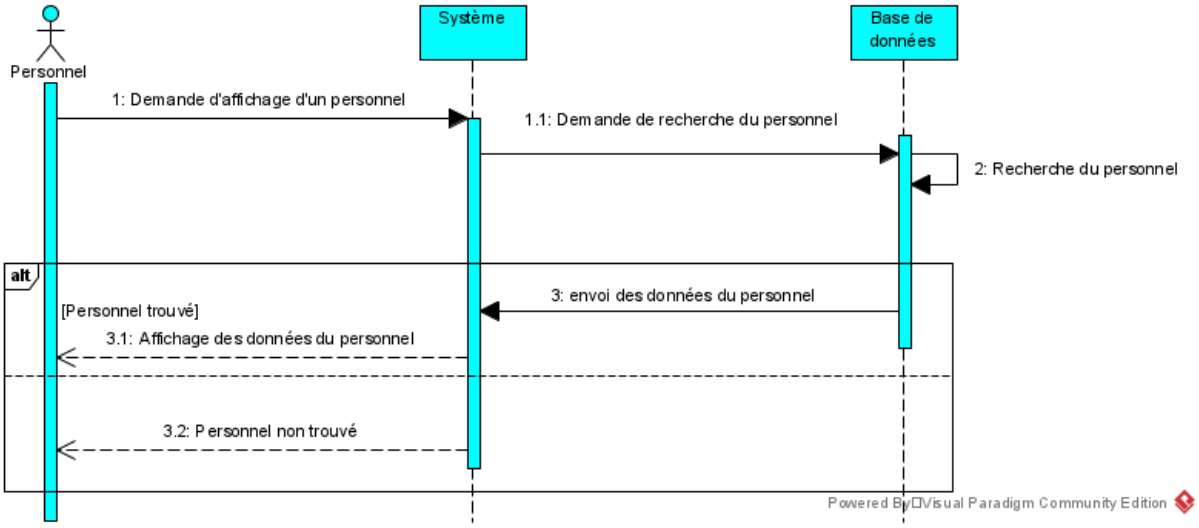
❖ Créer un personnel :



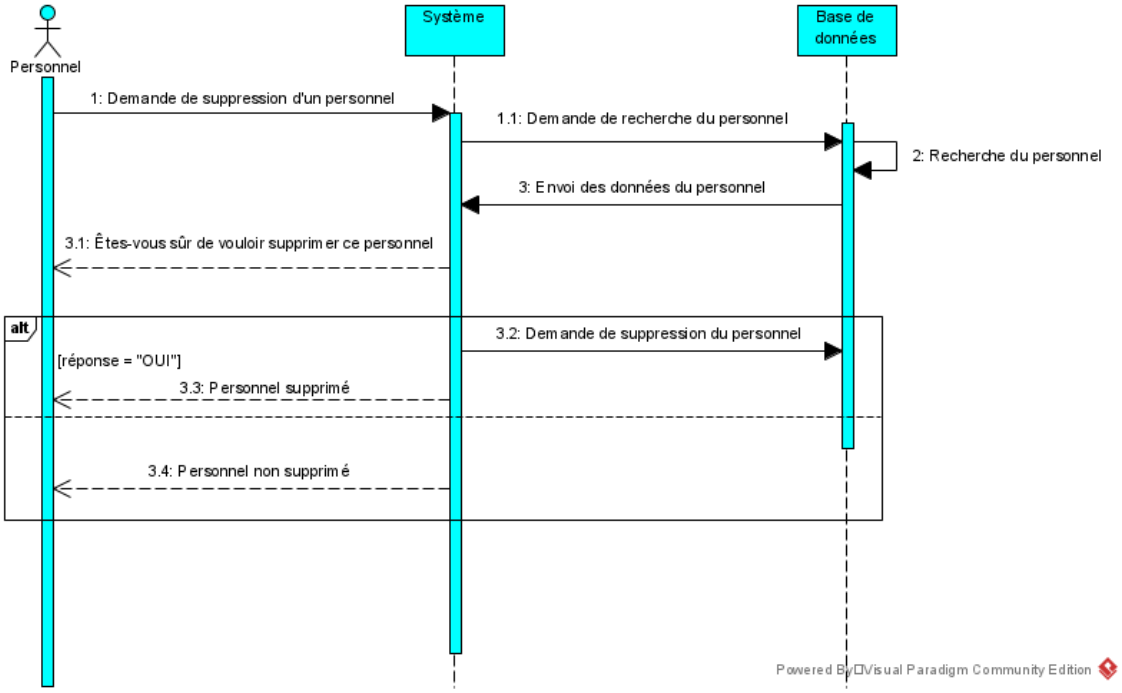
❖ Modifier un personnel :



❖ Afficher un personnel :



❖ Supprimer un personnel :



C. Explication des diagrammes :

Nous allons expliquer les diagrammes de séquences un par un :

- Création d'un personnel :
  Pour créer un personnel le système affiche un formulaire qui contiendras les différents champs nécessaires pour ajouter un personnel dans la base de données, après avoir rempli ce formulaire le système vérifie la validité du formulaire, s'il est validé le formulaire sera envoyé vers la base de données ou il y'auras une dernière vérification pour vérifier si le personnel existe déjà.

- Modification d'un personnel :
  Pour modifier un personnel le système affiche un formulaire qui contiendras les différents champs nécessaires pour modifier un personnel dans la base de données, après avoir rempli ce formulaire le système vérifie la validité du formulaire, s'il est validé le formulaire sera envoyé vers la base de données.

- Affichage d'un personnel :
  Pour afficher un personnel le système demande à la base de données d'afficher une liste contenant tout le personnel, puis le système demande à l'utilisateur de sélectionner un personnel, par la suite le système affiche les données du personnel sélectionné.

- Suppression d'un personnel :
  Pour supprimer un personnel le système demande à la base de données d'afficher une liste contenant tout le personnel, puis le système demande à l'utilisateur de sélectionner un personnel, par la suite le système demande à l'utilisateur de confirmer la suppression du personnel sélectionné, si l'utilisateur confirme la suppression, le système ordonne à la base de données de supprimer le personnel de la base de données.

3. Diagramme d'activité :

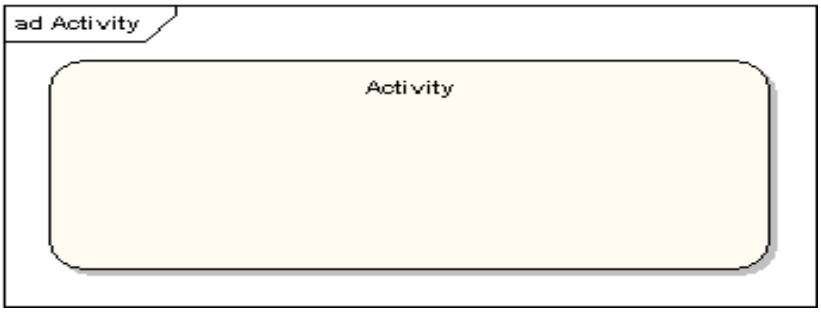
A. Qu'est-ce qu'un diagramme d'activité ?

- Diagramme d'activité est utilisé pour afficher la séquence des activités. Les diagrammes d'activité représentent le flux de travail à partir d'un point de départ au point d'arrivée. Détaillant les nombreux sentiers de décision, qui existent dans la progression des événements contenus dans l'activité. Ils peuvent être utilisés à des situations de détail, où le traitement parallèle peut survenir dans l'exécution de certaines activités. Les diagrammes d'activités sont utiles pour la modélisation d'entreprise où ils sont utilisés pour détailler les processus impliqués dans des activités commerciales.

- **Ces éléments se composent :**

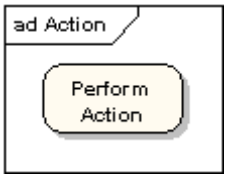
Activités :

Une activité est la spécification d'une séquence paramétrée de comportement. Une activité est représentée comme un rectangle à coins arrondis enfermant toutes les actions, les flux de contrôle et d'autres éléments qui composent l'activité.



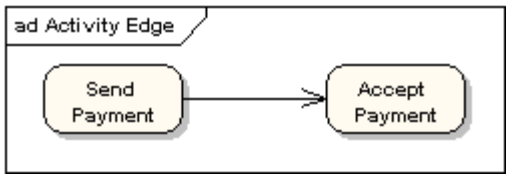
Actions :

Une action représente un pas (une étape) seul (simple) dans une activité. Les actions sont dénotées par des rectangles ronds-coincés.



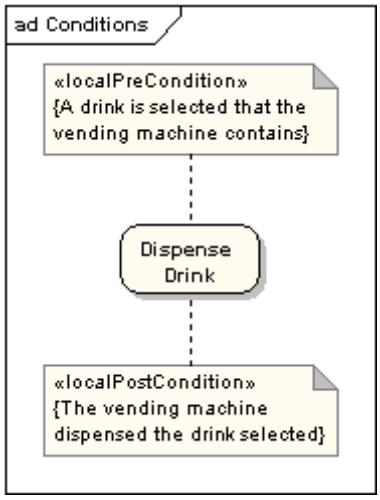
Flux de Contrôle :

Un Flux de Contrôle montre le flux de contrôle d'une action au prochain. Sa notation est une ligne avec une pointe de flèche.



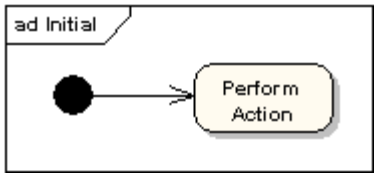
Contraintes d'Action :

Les contraintes peuvent être attachées à une action. Le diagramme suivant montre une action avec local pré et des post conditions.



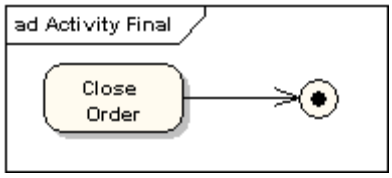
Nœud Initial :

Un initial ou le nœud de début est dépeint par un grand point noir, comme indiqué ci-dessous.

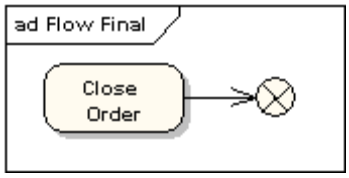


Nœud Final :

Il y a deux types de Nœud Final: activité et nœuds de finale de flux. Le nœud de finale d'activité est dépeint comme un cercle avec un point à l'intérieur.



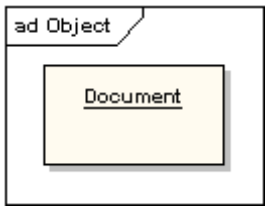
Le nœud de finale d'activité est dépeint comme un cercle avec un point à l'intérieur.



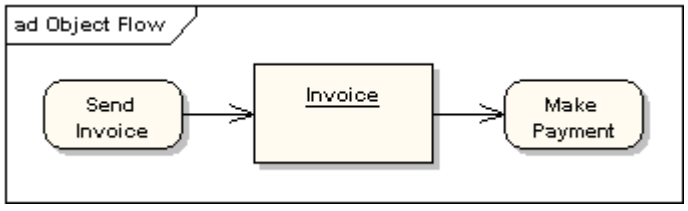
La différence entre les deux types de nœud est que le nœud de finale de flux dénote la fin d'un flux de contrôle seul; le nœud de finale d'activité dénote la fin de tous les flux de contrôle dans l'activité.

Objets et Flux d'Objet :

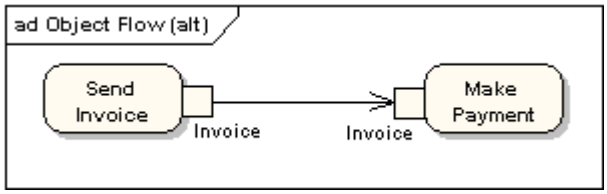
Un flux d'objet est un chemin le long lequel les objets ou des données peuvent passer. On montre un objet comme un rectangle.



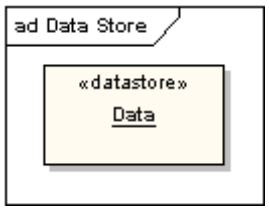
On montre un flux d'objet comme un connecteur avec une pointe de flèche dénotant la direction on passe l'objet.



Un flux d'objet doit avoir un objet sur au moins une de ses fins. Une notation de raccourci pour le susdit diagramme devrait utiliser des épingles de production et l'apport.

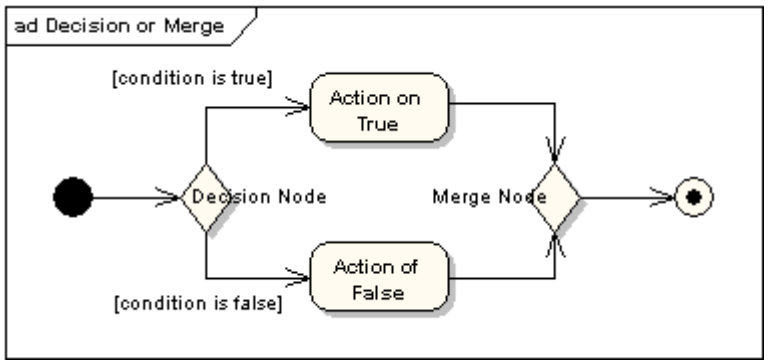


Un magasin de données est représenté comme un objet avec le « data store » mot-clé.



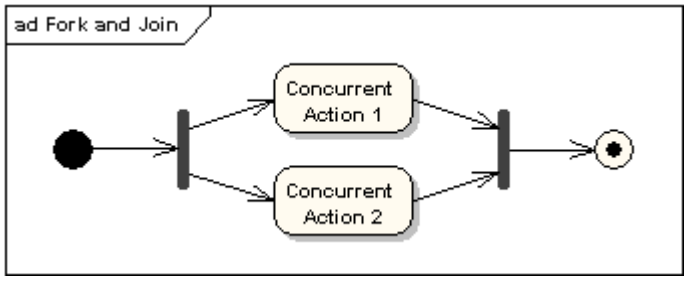
Décision et fusion de Nœuds :

Les nouds de décision et de fusionner les nouds ont la même notation: une forme de diamant. Ils peuvent tous deux être nommés. Les flux de contrôle à venir loin d'un noud de décision auront des conditions de garde qui permettront le contrôle de circuler si la condition de garde est satisfaite. Le schéma suivant illustre l'utilisation d'un nœud de décision et un noud de fusion.



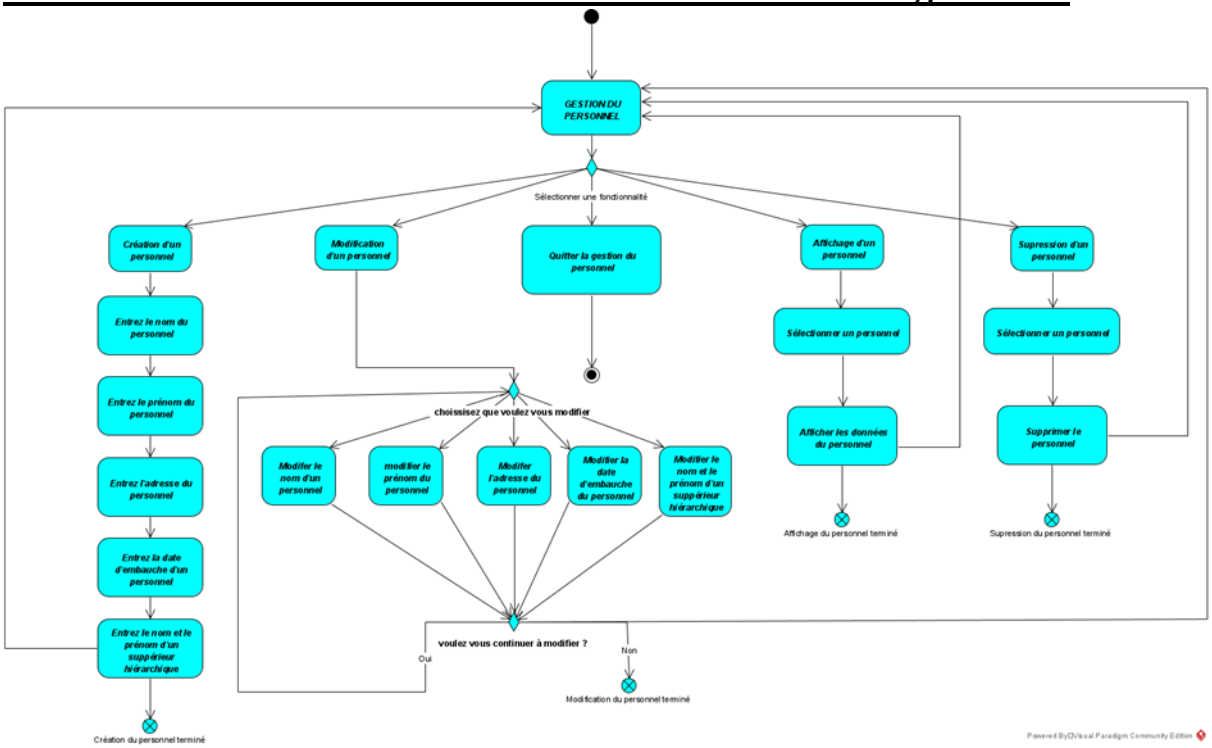
Fourchette et Joint Nœuds :

Fourches et Jointures ont la même notation: soit une barre horizontale ou verticale (l'orientation dépend du fait que le flux de contrôle est en cours d'exécution de gauche à droite ou de haut en bas). Ils indiquent le début et la fin des fils simultanés de contrôle. Le diagramme suivant montre un exemple de leur utilisation.



Une jointure est différente de la fusion en ce que la jointure synchronise deux entrées et produit une seule sortie. La sortie d'un rejoindre ne peut pas exécuter jusqu'à ce que toutes les entrées aient été reçues. Une fusion transmet les flux de contrôle droit à travers cela. Si deux ou plusieurs entrées sont reçus par un symbole de fusion. L'action pointée par sa sortie est exécutée deux fois ou plus.

B. Le diagramme :



C. Explication du diagramme :

Le diagramme commence par la première activité qui est la gestion du processus voulu dans ce cas-là il s'agit de la gestion du personnel, par la suite l'utilisateur peut choisir une des fonctionnalités de la gestion du personnel ou bien il peut quitter cette dernière, par exemple supposons qu'il a choisi l'affichage d'un personnel, il devra par la suite sélectionner un personnel et les données d'un personnel vont s'afficher juste après cette fonctionnalité se termine et l'utilisateur reviendra dans le menu de la gestion du personnel et pourra présélectionner une de ses fonctionnalités

4. Diagramme de classe :

A. Qu’est-ce qu’un diagramme de classes ?

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci. Ce diagramme fait partie de la partie statique d'UML car il fait abstraction des aspects temporels et dynamiques.

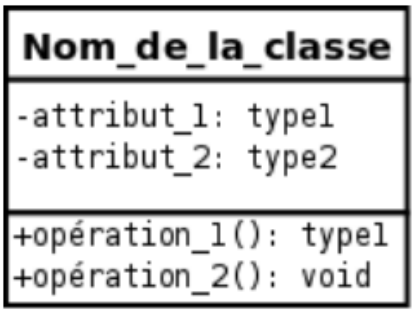
Une classe décrit les responsabilités, le comportement et le type d'un ensemble d'objets. Les éléments de cet ensemble sont les instances de la classe.

Une classe est un ensemble de fonctions et de données (attributs) qui sont liées ensemble par un champ sémantique. Les classes sont utilisées dans la programmation orientée objet. Elles permettent de modéliser un programme et ainsi de découper une tâche complexe en plusieurs petits travaux simples.

Les classes peuvent être liées entre elles grâce au mécanisme d'héritage qui permet de mettre en évidence des relations de parenté. D'autres relations sont possibles entre des classes, chacune de ces relations est représentée par un arc spécifique dans le diagramme de classes.

Elles sont finalement instanciées pour créer des objets (une classe est un *moule à objet* : elle décrit les caractéristiques des objets, les objets contiennent leurs valeurs propres pour chacune de ces caractéristiques lorsqu'ils sont instanciés).

Représentation graphique d’une classe :



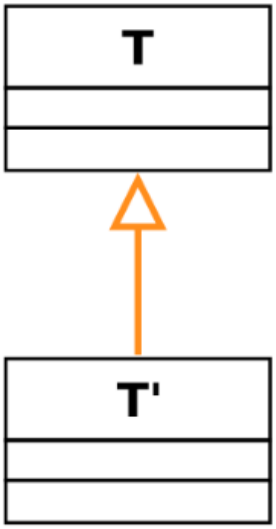
Une classe est un classeur. Elle est représentée par un rectangle divisé en trois à cinq compartiments.

Le premier indique le nom de la classe, le deuxième ses attributs et le troisième ses opérations. Un compartiment des responsabilités peut être

ajouté pour énumérer l'ensemble de tâches devant être assurées par la classe, mais pour lesquelles on ne dispose pas encore assez d'informations. Un compartiment des exceptions peut également être ajouté pour énumérer les situations exceptionnelles devant être gérées par la classe.

Représentation des différentes associations entre les classes :

Héritage :



L'héritage est un principe de division par généralisation et spécialisation, représenté par un trait reliant les deux classes et dont l'extrémité du côté de la classe mère comporte un triangle.

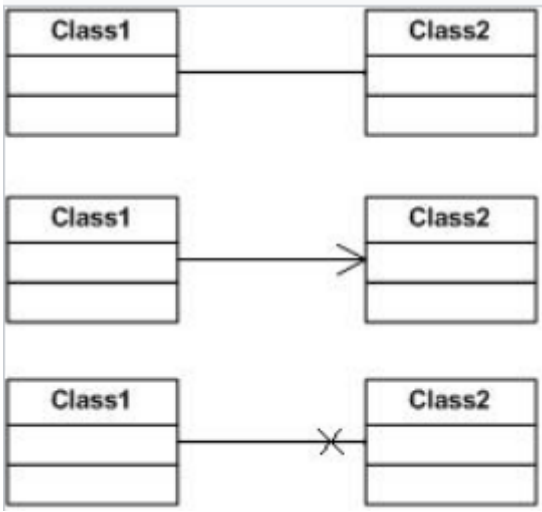
La classe fille hérite de tous les attributs et méthodes, qu'ils soient publics, protégés ou privés. Cependant, elle ne peut pas utiliser directement les attributs et méthodes privés (que ce soit en lecture ou en écriture), sauf par l'intermédiaire d'une méthode héritée (publique ou protégée).

Association :

L'association est une connexion sémantique entre deux classes (relation logique). Une association peut être nommée. L'invocation d'une méthode est une association. Elle peut être binaire, dans ce cas elle est représentée par un simple trait, ou n'aire, les classes sont reliées à un losange par des traits simples. Ces relations peuvent être nommées. L'association n'est utilisée que dans les diagrammes de classe.

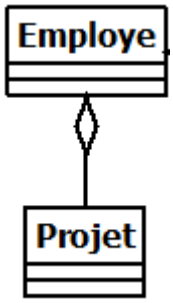
Multiplicité : comparable aux cardinalités du système Merise, sert à compter le nombre minimum et maximum d'instances de chaque classe dans la relation liant 2 ou plusieurs classes.

Navigabilité : indique si on pourra accéder d'une classe à l'autre. Si la relation est entre les classes A et B et que seulement B est navigable, alors on pourra accéder à B à partir de A mais pas réciproquement. Par défaut, la navigabilité est dans les 2 sens.



Navigabilité des associations
1- Bidirectionnelle
2- Mono-directionnelle, Invocation de méthode
3- interdit une association.

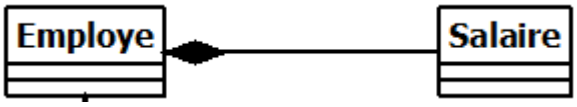
Agrégation :



L'agrégation est une association avec relation de subordination, représentée par un trait reliant les deux classes et dont l'origine se distingue de l'autre extrémité (la classe subordonnée) par un losange vide. Une des classes regroupe d'autres classes. L'objet T utilise une instance de la classe T'.

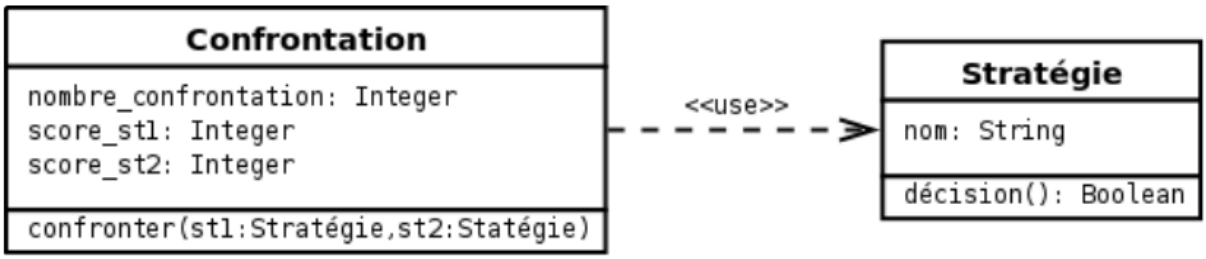
Composition :

La composition est une agrégation avec cycle de vie dépendant : la classe composée est détruite lorsque la classe mère disparait. L'origine de cette association est représentée par un losange plein.

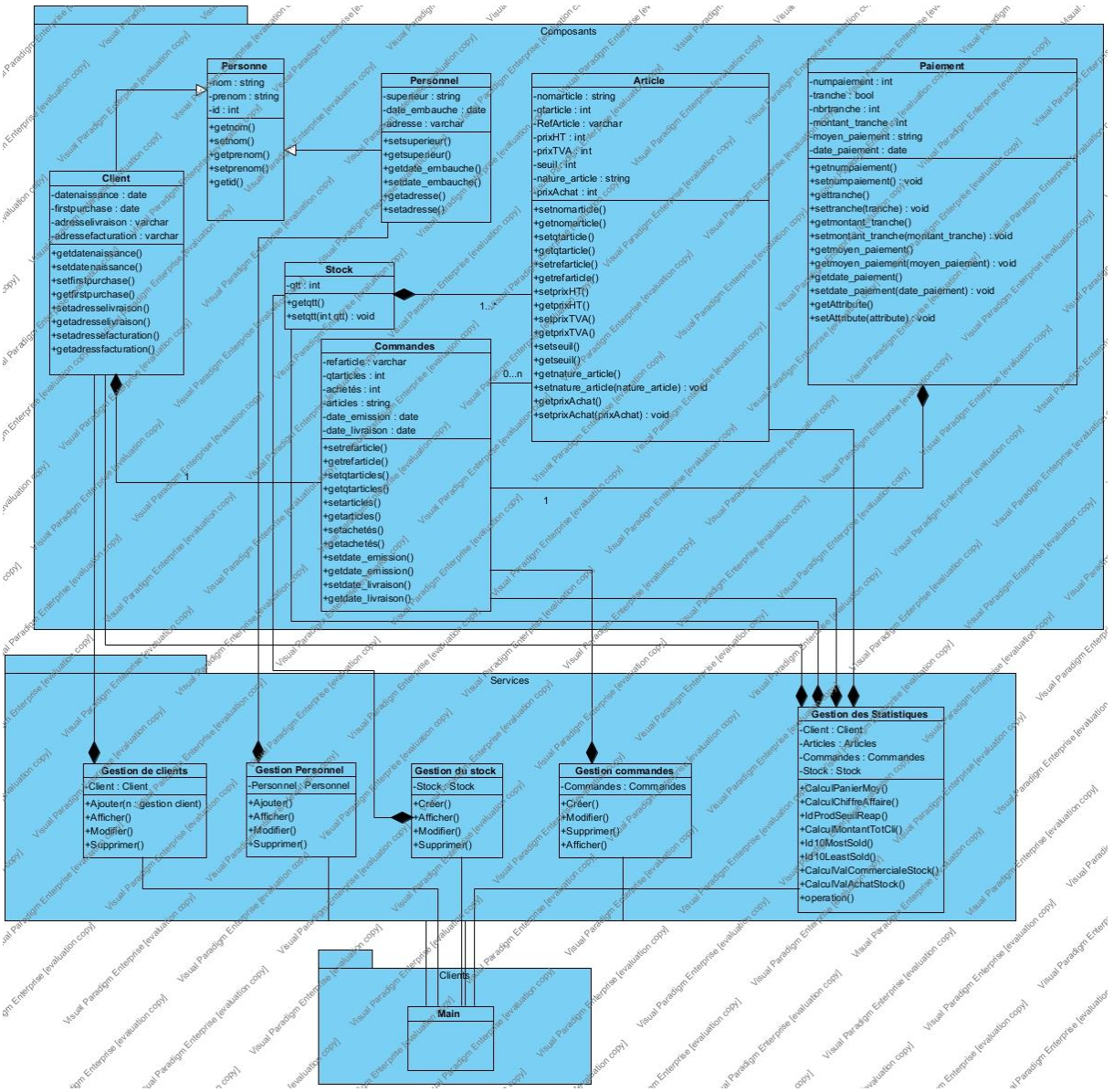


Dépendance :

Implique qu'une ou plusieurs méthodes reçoivent un objet d'un type d'une autre classe. Il n'y a pas de liaison en ce qui concerne la destruction d'objets mais une dépendance est quand même là. Elle est symbolisée par une flèche en pointillés, dont son extrémité possède trois traits qui se coupent en un même point.



B. Le diagramme :



C. Explication du diagramme :

Dans ce diagramme nous avons mis les classes que nous avons prévu de mettre dans le code du programme, Et comme vous voyez nous avons montré les différents liens entre les classes ainsi que les paramètres (propriétés et méthodes) de chaque classe. Et grâce à ce diagramme on comprend mieux l'aperçu général des schémas de notre application.

II. Environnement de développement opérationnel :

1. Visual Studio :

A. Qu'est-ce que Visual Studio ?

Microsoft Visual Studio est une suite de logiciels de développement pour Windows et mac OS conçue par Microsoft. La dernière version s'appelle Visual Studio 2019.

Visual Studio est un ensemble complet d'outils de développement permettant de générer des applications web ASP.NET, des services web XML, des applications bureautiques et des applications mobiles. Visual Basic, Visual C++, Visual C# utilisent tous le même environnement de développement intégré (IDE), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages. Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du Framework .NET, qui fournit un accès à des technologies clés simplifiant le développement d'applications web ASP et de services web XML grâce à Visual Web Developer.

B. Préparation du Framework VS :

Notre équipe chargée du projet utilisera Visual Studio pour le développement de l'application demandé, le Framework est prêt et installé et opérationnel.

2. Git Hub :

A. Qu’est-ce que Git Hub ?

Git Hub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. Ce site est développé en Ruby on Rails et Erlang par Chris Wanstrath, PJ Hyett et Tom Preston-Werner. Git Hub propose des comptes professionnels payants, ainsi que des comptes gratuits pour les projets de logiciels libres. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki pour chaque projet.

B. Préparation du service Git Hub ?

Afin d’assurer le bon déroulement de l’application et des échanges entre nous, nous allons utiliser Git Hub afin d’héberger notre programme chaque fois qu’on avance sur ce dernier, et ce qui nous permettras à tous de consulter l’avancement de l’application et de consulter sa dernière version et de le modifier que ce soit chacun de son côté ou en groupe.

3. SQL Server :

A. Qu’est-ce que SQL Server ?

Microsoft SQL Server est un système de gestion de base de données (SGBD) en langage SQL incorporant entre autres un SGBDR (SGBD relationnel ») développé et commercialisé par la société Microsoft. Il fonctionne sous les OS Windows et Linux (depuis mars 2016), mais il est possible de le lancer sur Mac OS via Docker, car il en existe une version en téléchargement sur le site de Microsoft.

B. Préparation du SGBDR SQL Server ?

Pour la réalisation de l’application demandée il est nécessaire que notre programme soit connecté avec une base de données afin de communiquer avec cette dernière, nous avons donc choisis d’utiliser SQL Server pour la création de notre base de données.

Conclusion :

Nous avons réalisé les deux premières parties du projet qui consistait à modéliser le programme et préparer notre environnement de travail, désormais nous pouvons passer à la partie création de la base de données et la partie code.