21 DECEMBRE 2023

PROJET BASE DE DONNEES

SYSTEME D'INFORMATION POUR UN LABORATOIRE D'ANALYSE MEDICALE

MATHIS DEFYN & VALENTIN PERROT
BACHELOR 2 DIA

Table des matières

I. Introduction

L'objectif de ce projet est de créer une base de données d'un centre médical dans notre cas un laboratoire d'analyse. Le sujet concerne la conception et la réalisation du système d'information de gestion des rendez-vous et des dossiers des patients pour le besoin d'un laboratoire d'analyse médicale.

La base de données à créer doit permettre de recenser les différents renseignements nécessaires au fonctionnement du laboratoire. Ces renseignements concernent, par exemple, les différents clients, les analyses médicales ou encore les prisse de rendez-vous.

Dans le cahier des charges de la base de données, on nous demande de gérer :

- La gestion des rendez-vous
- La gestion des patients
- La gestion de la facturation
- Reporting
- Messagerie

La manipulation de la base de données doit être permise par un exécutable. Un médecin, un patient ou une réceptionniste ne doivent pas avoir à manipuler directement les tables.

II. Analyse des besoins

Les besoins demander étaient multiples, il y a déjà la gestion des rendez-vous pour offrir la possibilité :

• Au patient de demander, modifier, annuler ou confirmer son rendez-vous

• Au réceptionniste d'ajouter, modifier ou annuler un rendez-vous

• Au médecin et réceptionniste de visualiser le flux des rendez-vous (rendez-vous en cours, futures, achevés)

Avec quelques règles métier :

• Un rendez-vous peut être pris par le patient ou parla réceptionniste si le patient se présente surplace

• La prise d'un nouveau rendez-vous, le créneau pris devient inaccessible aux autres patients

• Etats d'un rendez-vous : Nouveau, confirmé, en cours, achevé, annulé

Nous avons ensuite la gestion des patients pour donner l'opportunité :

• Au patient de gérer son compte (informations générales, coordonnées, assurance médicale, historique des analyses(optionnelle)…)

• Au réceptionniste de gérer le compte d'un patient

• Au médecin de gérer le dossier médical d'un patient, les visites médicales et les résultats d'analyses.

Avec c'est règles métier :

• Le médecin doit avoir la possibilité de visualiser l'historique d'un patient ainsi que mettre à jour son dossier médical avec de nouvelles analyses (résultats, commentaires…)

Puis il y a aussi la gestion de la facturation pour offrir la possibilité :

• Au réceptionniste de renseigner le montant d'une visite, gérer les factures clients etc.

• Au médecin de faire le suivi des recettes journalières, mensuelles etc.

• Au patient de visualiser les factures de ses consultations (analyses…)

Avec comme règles :

• La réceptionniste doit avoir la possibilité d'imprimer la facture d'un patient…

Et il y a le reporting où le système devra permettre la visualisation des flux des rendez-vous, profil et historique de visites d'un patient, dossier médical d'un patient avec ses analyses avec c'est résultats et les recettes

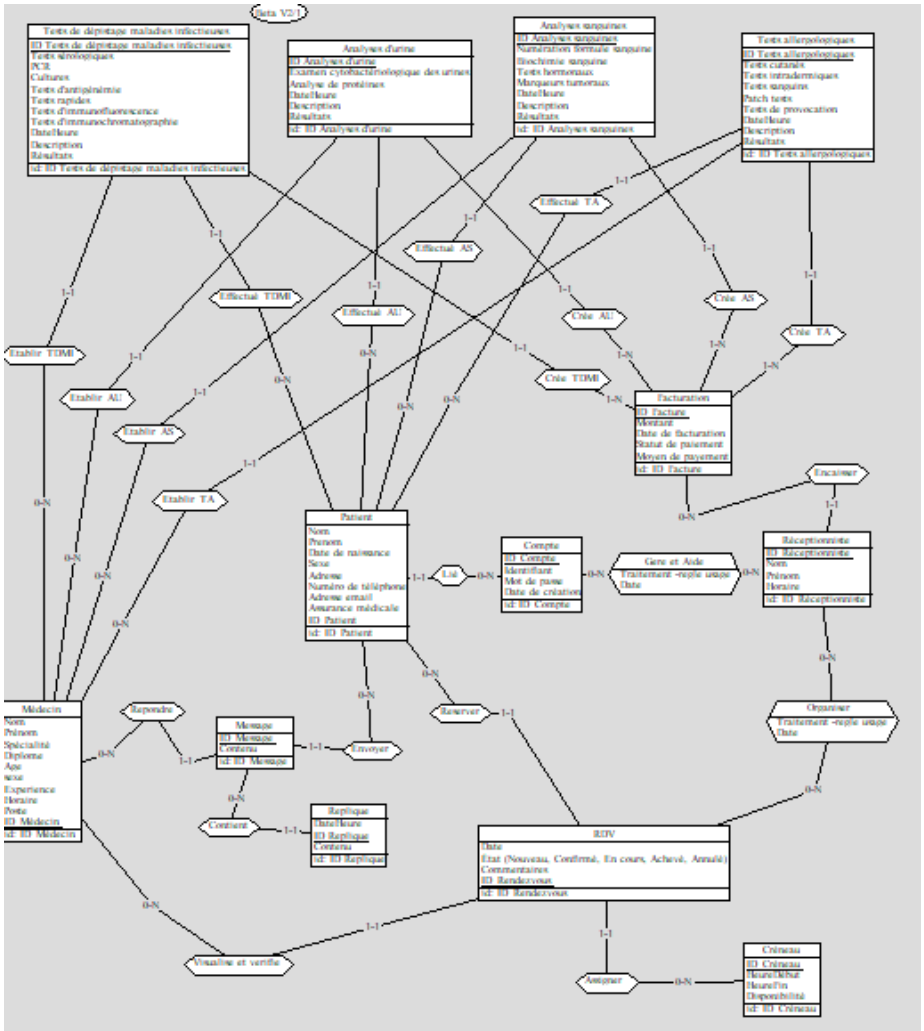
Et enfin la gestion de la messagerie pour donner la possibilité au patient d'écrire un message à son médecin traitant, au médecin de répondre aux messages de ses patients

III. Conception de la base de données

Au travers de ce projet. Nous avons pu créer une base de données d'un laboratoire médical et plus particulièrement de cas d'un laboratoire d'analyse médicale que nous avons nommé Bouboucenter. Pour ce faire, nous avons fait cela en plusieurs étapes dont la première étant la conception d'un Mcd. (Modèle conceptuel de données). Ensuite, grâce à différents logiciels donc, on vous parlera plus tard On pourra obtenir la suite logique des choses, c'est à dire 1 MLD et le SQL.

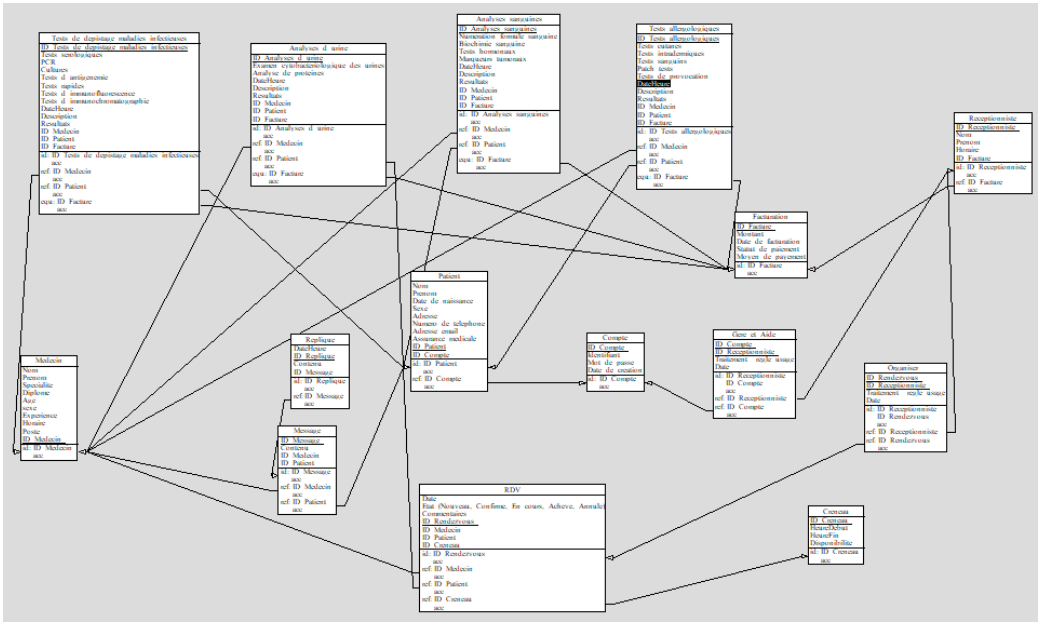
Nous avons en dessous 3 images. Représentant respectivement le MCD de sa première partie, la 2e étant le MLD et la 3e représentant le SQL généré par ce MLD

1) Modèle conceptuel de données



Lors de ce projet qui nous a poussé au travers de notre réflexion, à s'imaginer un véritable cabinet médical ici, dans notre cas, un laboratoire d'analyse médicale, cela nous a fait réfléchir à différentes règles de gestion que nous allons vous énumérer maintenant :

- La première règle de gestion est la suivante : Une personne n'a et n'aura qu'un seul compte, mais ce même compte peut avoir plusieurs Patients. En effet, dans le cas où une famille possède un compte, les enfants n'ont pas la capacité de se créer des comptes, donc nous avons réfléchi et dit que les enfants seraient sur le compte parental, ce qui nous donne cette relation dans la MCD.
- On considère aussi qu'un rendez-vous n'est pas seul sur un même créneau horaire. En effet, au vu du nombre de médecins ont serré dans une base de données, on considère que plusieurs rendez-vous sont possibles en même temps
- On estime aussi qu'un médecin peut visualiser les rendez-vous des autres médecins et donc aussi des autres Patients.
- Concernant les réceptionnistes, on considère dans ce MCD qu'elles peuvent organiser à plusieurs le traitement et les règles d'usage, ou encore la date du rendez-vous. Au cas où une a commencé mais n'est pas en État ou n'a pas la possibilité de terminer ce changement. Ou le même dans le cas d'un problème technique qui résulterait en l'impossibilité de modification de la part d'une réceptionniste.
- Pour les médecins, on considère que ces derniers peuvent être à plusieurs sur une même conversation, afin de répondre à un seul patient.
  En effet, dans le cas où ce sont 2 médecins qui font des analyses ayant le même but, par exemple. Un test sanguin et un test d'urine afin de vérifier le taux de sucre dans le sang, Alors il peut être nécessaire que les deux médecins ayant fait les analyses, dans le cadre ou chacune analyse relève de leur compétence personnelle ait le patient en même temps afin de répondre de façon claire et précise aux patients.
- On considère qu'un médecin peut établir plusieurs analyses médicales, mais qu'une analyse médicale à la possibilité de ne peut pas être établie que par un seul médecin. En effet, on considère que les médecins de notre laboratoire sont spécialisés et que chacun a ses spécialités propres et qu'ils ne peuvent pas faire des analyses conjointes au vu de leurs compétences différentes.



2) Le dictionnaire des données

NOM	Description	Type
ID_Patient	Nom d'identification du patient	Varchar
Nom	Nom du patient	Varchar
Prenom	Prenom du patient	Varchar
Date de naissance	Date de naissance du patient	Date
Sexe	Sexe du patient	Varchar
Adresse	Adresse du patient	Varchar
Numéro de Téléphone	Numéro du patient	Varchar
Adresse email	Email du patient	Varchar
Assurance médicale	Assurance médicale du patient	Numeric
ID_Compte	Nom d'identification du compte	Varchar
Identifiant	Identifiant du compte	Varchar
Mot de passe	Mot de passe du compte	Varchar

Date de création	Date de création du compte	Date
ID_Receptionniste	Nom d'identification du Receptionniste	Varchar
Nom	Nom du Receptionniste	Varchar
Prenom	Prenom du Receptionniste	Varchar
Horaire	Horaire du Receptionniste	Date
ID_Facture	Nom d'identification de la Facture	Varchar
Montant	Montant de la Facture	Numeric
Date de facturation	Date de facturation de la Facture	Date
Statut de paiement	Statut de paiement de la Facture	Varchar
Moyen de payement	Moyen de payement de la facture	Varchar
ID_Rendezvous	Nom d'identification du Rendez-vous	Varchar
Date	Date du rendez-vous	Date
Etat	Etat du rendez-vous	Varchar
Commentaires	Commentaire du rendez-vous	Varchar
ID_Créneau	Nom d'identification du Créneau	Varchar
HeureDébut	Heure de début du Créneau	Date
HeureFin	Heure de fin du Créneau	Date
Disponibilité	Disponibilité du Créneau	Date
ID_Message	Nom d'identification du Message	Varchar
Contenu	Contenu du message	Varchar
ID_Replique	Nom d'identification de la replique	Varchar
Contenu	Contenu de la replique	Varchar
DateHeure	Date et Heure de la replique	Date
ID_Médecin	Nom d'identification du médecin	Varchar
Nom	Nom du médecin	Varchar
Prenom	Prenom du médecin	Varchar
Spécialité	Spécialité du médecin	Varchar

Diplome	Diplome du médecin	Varchar
Age	Age du médecin	Numeric
sexe	Sexe du médecin	Varchar
Experience	Experience du médecin	Varchar
Horaire	Horaire du médecin	Date
Poste	Poste du médecin	Varchar
ID_Tests_de_dépistage maladies_infectieuses	Nom d'identification du Tests de dépistage maladies infectieuses	Varchar
Tests sérologiques	Tests sérologiques	Varchar
PCR	PCR	Varchar
Cultures	Cultures	Varchar
Tests d'antigénémie	Tests d'antigénémie	Varchar
Tests rapides	Tests rapides	Varchar
Tests d'immunofluorescence	Tests d'immunofluorescence	Varchar
Tests d'immunochromatographie	Tests d'immunochromatographie	Varchar
DateHeure	Date et Heure test maladies infectieuses	Date
Description	Description test maladies infectieuses	Varchar
Résultats	Résultats test maladies infectieuses	Varchar
ID_Analyses_d'urine	Nom d'identification de Analyses d'urine	Varchar
Examen cytobactériologique des urines	Examen cytobactériologique des urines de Analyses d'urine	Varchar
Analyse de protéines	Analyse de protéines de Analyses d'urine	Varchar
DateHeure	Date et Heure test maladies infectieuses	Date
Description	Description de Analyses d'urine	Varchar
Résultats	Résultats de Analyses d'urine	Varchar
ID_Analyses_sanguines	Nom d'identification de Analyses sanguines	Varchar

Numération formule sanguine	Numération formule sanguine de Analyses sanguines	Varchar
Biochimie sanguine	Biochimie sanguine de Analyses sanguines	Varchar
Tests hormonaux	Tests hormonaux de Analyses sanguines	Varchar
Marqueurs tumoraux	Marqueurs tumoraux de Analyses sanguines	Varchar
DateHeure	Date et Heure de Analyses sanguines	Date
Description	Description de Analyses sanguines	Varchar
Résultats	Résultats de Analyses sanguines	Varchar
ID_Tests_allergologiques	Nom d'identification du Tests allergologiques	Varchar
Tests cutanés	Tests cutanés du Tests allergologiques	Varchar
Tests intradermiques	Tests intradermiques du Tests allergologiques	Varchar
Tests sanguins	Tests sanguins du Tests allergologiques	Varchar
Patch tests	Patch tests du Tests allergologiques	Varchar
Tests de provocation	Tests de provocation du Tests allergologiques	Varchar
DateHeure	Date et Heure du Tests allergologiques	Date
Description	Description du Tests allergologiques	Varchar
Résultats	Résultats du Tests allergologiques	Varchar

3) Code SQL

```
87
88  create table Medecin (
89       Nom varchar(50) not null,
90       Prenom varchar(50) not null,
91       Specialite varchar(50) not null,
92       Diplome varchar(50) not null,
93       Age numeric(5,0) not null,
94       sexe varchar(50) not null,
95       Experience varchar(50) not null,
96       Horaire date not null,
97       Poste varchar(50) not null,
98       ID_Medecin_ numeric(10,0) not null,
99       constraint ID_Medecin_ID primary key (ID_Medecin_));
100
101 create table Organiser_ (
102      ID_Rendezvous_ numeric(10,0) not null,
103      ID_Receptionniste_ varchar(20) not null,
104      Traitement__regle_usage varchar(20) not null,
105      Date date not null,
106      foreign key (ID_Receptionniste_) references Receptionniste,
107      foreign key (ID_Rendezvous_) references RDV,
108      constraint ID_Organiser__ID primary key (ID_Receptionniste_, ID_Rendezvous_));
109
110
111 create table Patient (
112      Nom varchar(50) not null,
113      Prenom varchar(50) not null,
114      Date_de_naissance date not null,
115      Sexe varchar(1) not null,
116      Adresse varchar(50) not null,
117      Numero_de_telephone varchar(20) not null,
```

Après la création du MCD et la génération du MLD, DB main nous a généré un code SQL qui va nous permettre de construire notre base de données du laboratoire d'analyse médicale sur DB Browser. Toutes les entités du MCD vont devenir des noms de table avec ces attribués associé.

Nous avons du modifié un peu le code, car DB Browser n'accepter pas les "Alter table", nous avons du donc intégré les requêtes dans la création des tables. On peut le voir dans la photo du code SQL fournie où on intègre le "foreign key (ID_Rendezvous_) references RDV" dans la création de la table "Organiser_". Après avoir réintègre toutes les requêtes des "Alter table" la base de données à pu se crée.

IV. Implémentation technique

1) Fonction Python

Dans la suite du projet après avoir réussi à crée la base de données, il fallait maintenant remplir cette base de données donc nous avons crée plusieurs types de fonctions pour effectué différente manipulation dans la base de données.

```
def ajouter_medecin(medecin: Medecin):
    """ Fonction pour ajouter un médecin. """
    try:
            missing = 1
            for  data in medecin :
                    if  data[1] == "string" :
                      missing = 0
                      print("ahhhhhh")
                      return("Missing info")

            if missing == 0:
                print("error")
                conn.rollback()
                return("eh ben nsml")

            else:
                colonnes = ', '.join(medecin.model_dump().keys())
                placeholders = ', '.join('?' * len(medecin.model_dump()))
                requete = f"INSERT INTO Medecin ({colonnes}) VALUES ({placeholders})"
                cursor.execute(requete, list(medecin.model_dump().values()))
                conn.commit()

                return {"message": "Médecin ajouté avec succès."}
    except Exception as e:
            conn.rollback()
            raise HTTPException(status_code=500, detail=str(e))
```

Comme on a pu le voir, nous avons donc créé des fonctions qui ont pour but de rajouter ou d'ajouter des données afin de créer des personnes dans notre base de données comme nous pouvons voir dans la photo précédente peut être applicable à l'ensemble des entités de notre base de données. En effet, nous avons des réceptionnistes, des patients ou même différents types d'analyses qui, grâce à ce type de fonction, peuvent être inclus ou peuvent être modifiés ou peuvent être même créés en tant que personne dans la base de Données

```python
def modifier_patient(id_patient: str, patient: Patient):
    """ Fonction pour modifier un patient. """
    try:
        patient_data = patient.model_dump()
        del patient_data['ID_Patient']  # Assuming 'ID_Patient' should not be updated

        clause_set = ', '.join([f"{key} = :{key}" for key in patient_data.keys()])
        requete = f"UPDATE Patient SET {clause_set} WHERE ID_Patient = :ID_patient"
        patient_data['ID_patient'] = id_patient

        cursor.execute(requete, patient_data)
        affected_rows = cursor.rowcount
        conn.commit()

        if affected_rows == 0:
            return {"message": "No rows updated. Check if the ID exists."}
        else:
            return {"message": f"Successfully updated {affected_rows} row(s)."}
    except Exception as e:
        return {"error": str(e)}
```

Nous avons aussi crée des fonctions pour modifier des données de personnes ou modifier quelque chose comme par exemple modifier des données sur un médecin, un patient, une réceptionnistes ou sinon modifier le compte d'un patient et l'état d'un rendez-vous.

```python
def visualiser_rdv_medecin(id_medecin: str):
    """ Fonction pour visualiser les rendez-vous d'aujourd'hui pour un médecin. """
    try:
        date_aujourd_hui = datetime.now().strftime("%Y-%m-%d")
        requete = "SELECT * FROM RDV WHERE ID_Medecin_ = :id_medecin AND Date_RDV = :date_aujourd_hui"
        curseur.execute(requete, {'id_medecin': id_medecin, 'date_aujourd_hui': date_aujourd_hui})
        fl = curseur.fetchall()
        conn.commit()
        return fl

    except Exception as e:
        return {"error": str(e)}
```

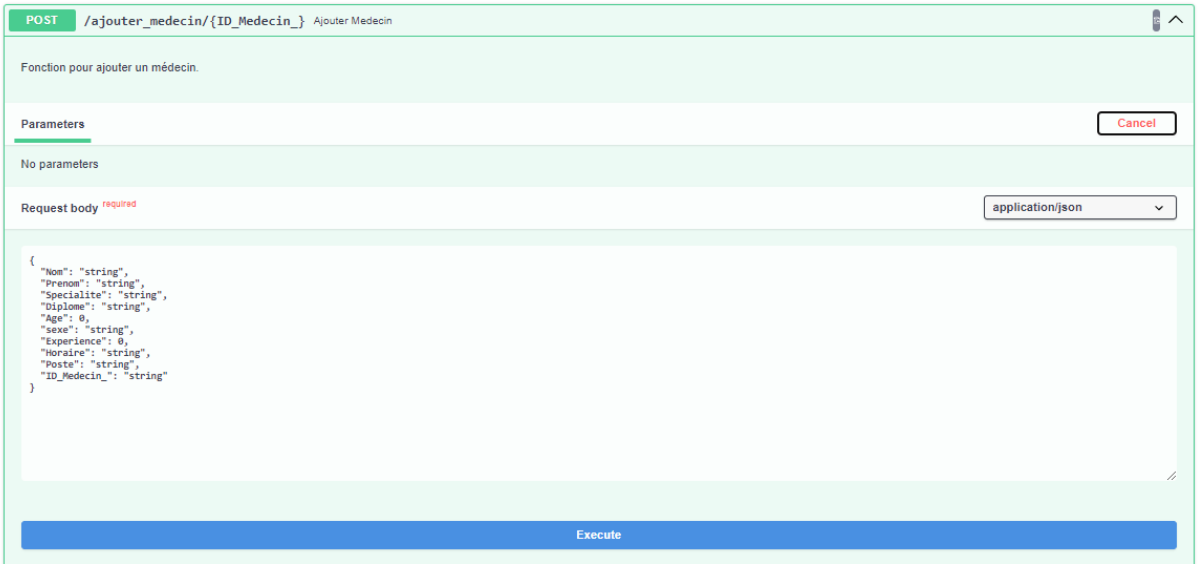
Nous avons ensuite crée des fonctions pour visualiser par exemple les rendez-vous d'un médecin dans une journée, le dossier d'un patient ou l'état d'un rendez-vous.

```python
def supprimer_M_P_R(id_element:str, table):
    """ Fonction pour supprimer un Medecin, un Patient, ou une Receptionniste. """
    try:
        tables_autorisees = ['Medecin', 'Patient', 'Receptionniste']
        if table not in tables_autorisees:
            raise ValueError(f"La table spécifiée '{table}' n'est pas autorisée. Choisissez parmi {tables_autorisees}.")
        else :
            if table == "Medecin":
                requete = f"DELETE FROM {table} WHERE ID_Medecin_ = :id"
                cursor.execute(requete, {'id': id_element})
                return {"message": "Medecin supprimer avec succès."}

            elif table == "Patient":
                requete = f"DELETE FROM {table} WHERE ID_Patient = :id"
                cursor.execute(requete, {'id': id_element})
                return {"message": "Patien supprimer avec succès."}

            else:
                requete = f"DELETE FROM {table} WHERE ID_Receptionniste_ = :id"
                cursor.execute(requete, {'id': id_element})
                return {"message": "Receptionniste supprimer avec succès."}
            conn.commit()
    except Exception as e:
        return {"error": str(e)}
```

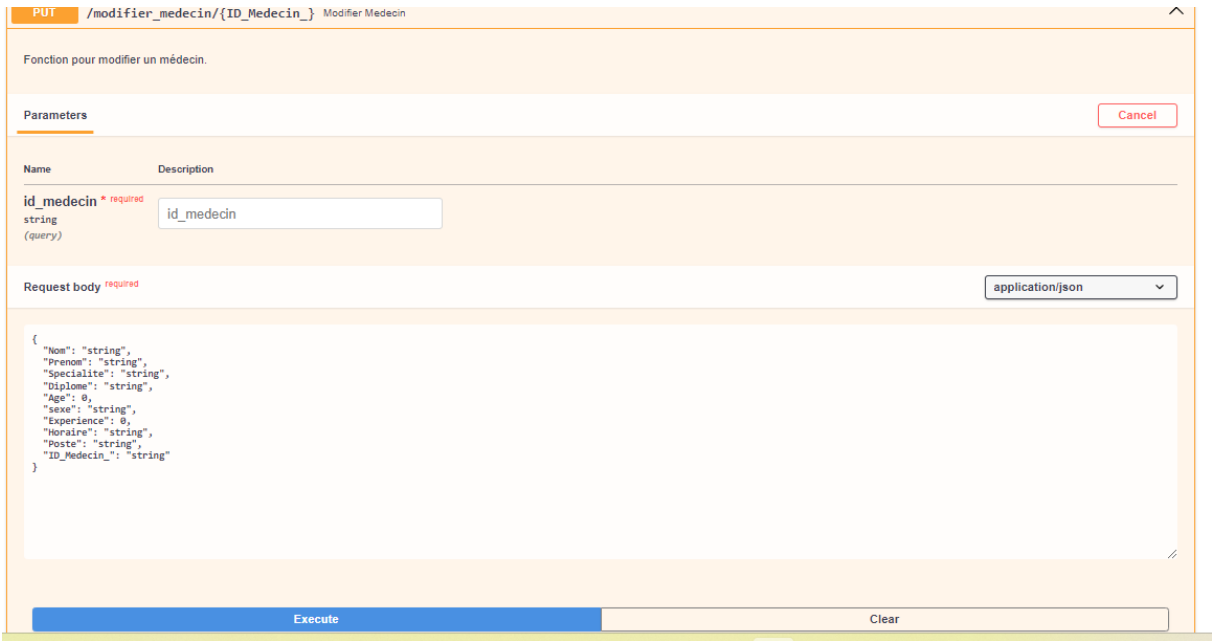
Mais aussi crée des fonctions pour supprimer les données de personnes comme par exemple supprimer le compte d'un patient et c'est données personnel ou sinon le données d'un médecin ou une réceptionniste qui son partie du laboratoire.

2) API

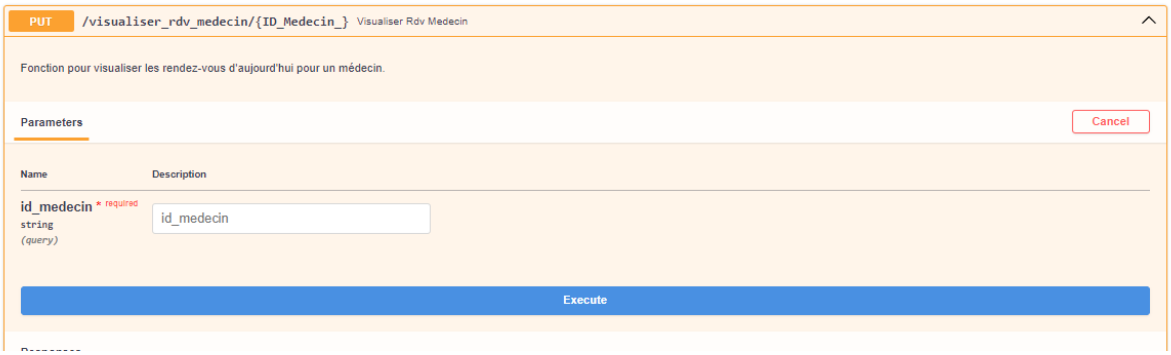
Maintenant que nous avons vu comment nous avons développé les fonctions la base de la future API. Nous allons maintenant parler du point suivant. En effet, nous avons déjà abordé les fonctions de la composante, mais maintenant nous allons la créer afin de remplir la base de données de façon autonome et indépendant des logiciels que nous avons au préalable utilisés.



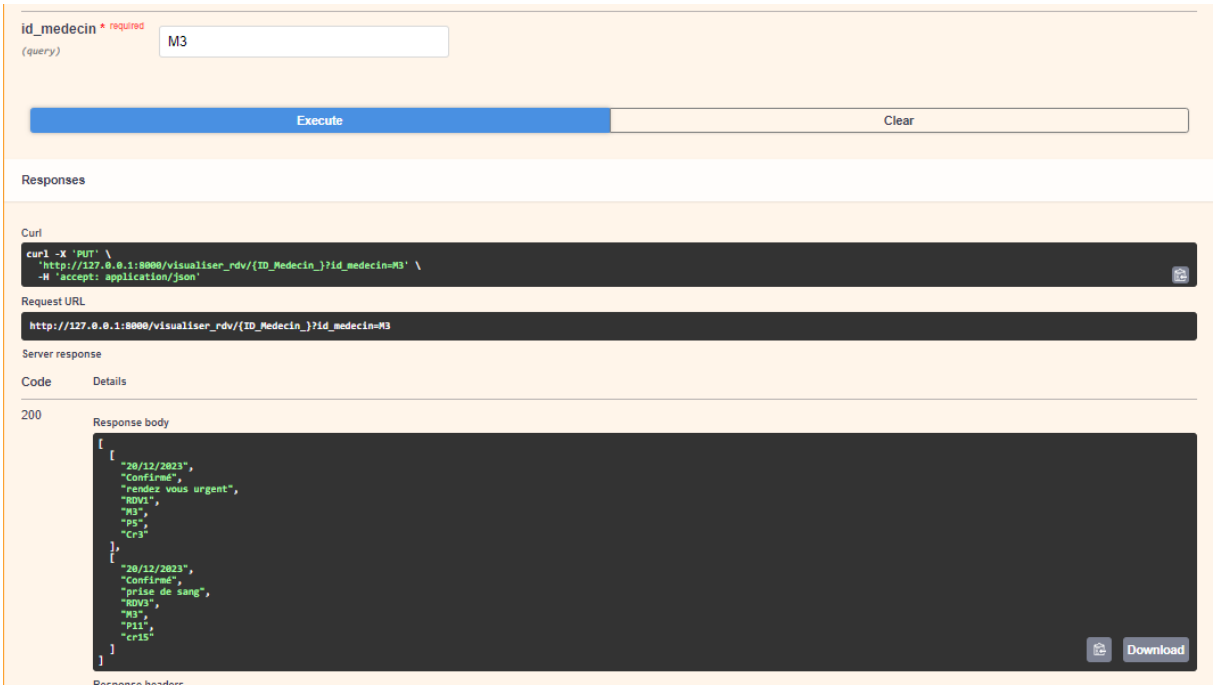
Sur cette image on voit l'exemple pour ajouter un médecin, on remplace dans le "Exemple Value" les "string" pour la chaine de caractère ou les "0" pour un nombre entier, par les informations sur le médecin qu'on veut rentrer dans la base de données puis on clique sur "Execute" pour ajouter.



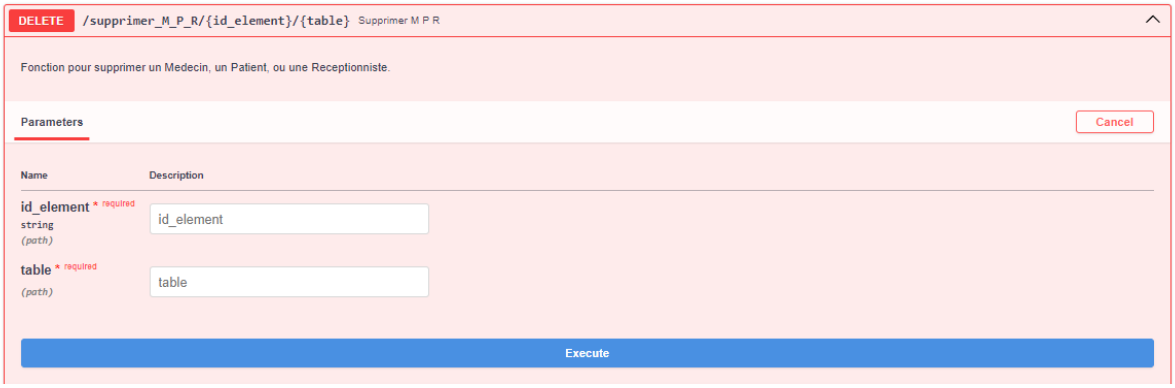
Ensuite sur cette image on voit l'exemple pour modifie les données d'un médecin, on tape Id_Medecin du médecin qu'on veut modifié ensuite on remplace dans le "Exemple Value" les "string" pour la chaine de caractère ou les "0" pour un nombre entier, par les informations sur le médecin déjà enregistré et les modifications qu'on veut rentrer dans la base de données puis on clique sur "Execute" pour modifié.



Puis sur cette image on voit l'exemple pour visualiser les rendez-vous d'aujourd'hui d'un médecin, on tape l'Id_Medecin du médecin puis on clique sur "Execute" pour visualiser tout les rendez-vous du jour.



Voici le resultat ci-dessus de visualiser rendez-vous d'aujourd'hui pour le medecin avec ID_medecin = M3.



Et enfin sur cette image on voit l'exemple pour supprimer les données d'un medecin, un patient ou une receptionniste. Pour cela on tape l'ID de la personne voulu et le nom de la table associé puis on clique sur "Execute" afin de l'enlever de la base de données.

3) Code HTML

Maintenant que nous avons vu l'API, le MCD, le MLD et enfin le SQL, nous allons vous parler de l'interface graphique composée de HTML, de CSS et de Javascript que nous avons réalisé pour ce projet.

Pour ce faire, nous avons dû apprendre 3 nouveaux langages pour réaliser cette interphase graphique. Après avoir établi les nécessités pour faire l'interface graphique qui d'après nous, était compose d'une page d'accueil, Une page permettant la création de nouvelles entités dans la base de données ou encore la modification de ces dernières et enfin un agenda permettant à chacun et chacune de visualiser les rendez-vous futurs,

Pour ce faire, nous avons dans un premier temps tenté d'implémenter la même structure HTML pour chaque page. Ensuite, pour répondre à une question d'esthétisme, nous avons appliqué sur chaque page le même style CSS.

Ensuite nous avons intégré dans notre interface graphique base de données qui se traduit par la récupération et l'envoi de données via L'API.

Et enfin, nous avons fait une dernière chose qui est le test de l'interface graphique afin de répondre aux difficultés que nous avons surmontées tout au long de la création de cette interface.

V. Base de données

Donc maintenant et après avoir tout tester voici ci-desous un exemple de la base de données remplie pour la table Patient et Médecin:

Nom	Prenom	Date_de_naissance	Sexe	Adresse	Numero_de_telephone	Adresse_email	Assurance_medicale	ID_Patient	ID_Compte_
Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre
Dupuis	Julie	01/12/1970	F	90 Rue Exemple, Ville, Pays	0657983725	Dupuis.Julie@gmail.com	7541515425	P2	C2
Lefevre	Marc	16/06/2004	M	74 Rue Exemple, Ville, Pays	0710841831	Lefevre.Marc@outlook.com	7295513102	P3	C3
Martin	Isabelle	10/08/1987	F	64 Rue Exemple, Ville, Pays	0749872072	Isabelle.Martin@gmail.com	1288088755	P4	C4
Gagnon	Éric	08/02/1950	M	57 Rue Exemple, Ville, Pays	0633750305	Gagnon.Éric@orange.fr	7967660186	P5	C5
Rousseau	Sophie	23/03/1958	F	22 Rue Exemple, Ville, Pays	0761959098	Rousseau.Sophie@sfr.fr	4302436451	P6	C6
Beaulieu	François	19/05/1996	M	42 Rue Exemple, Ville, Pays	0246855661	Beaulieu.François@gmail.com	5113413836	P7	C7
Bergeron	Catherine	22/08/1989	F	67 Rue Exemple, Ville, Pays	0137560993	Bergeron.Catherine@outlook.com	8332573723	P8	C8
Fortin	Patrick	05/11/1954	M	48 Rue Exemple, Ville, Pays	0630400225	Fortin.Patrick@sfr.fr	1234064868	P9	C9
Roy	Caroline	06/11/1988	F	10 Rue Exemple, Ville, Pays	0951368313	Roy.Caroline@orange.fr	1325605739	P10	C10
Lavoie	Mathieu	03/03/1978	M	80 Rue Exemple, Ville, Pays	0724036034	Lavoie.Mathieu@outlook.com	2214170801	P11	C11
Leclerc	Nathalie	06/12/1964	F	26 Rue Exemple, Ville, Pays	0696666401	Leclerc.Nathalie@gmail.com	3882564946	P12	C12
Morin	Alexandre	14/03/1956	M	75 Rue Exemple, Ville, Pays	0242789569	Morin.Alexandre@outlook.com	5468930067	P13	C13
Pelletier	Geneviève	17/05/1972	F	11 Rue Exemple, Ville, Pays	0166752120	Pelletier.Geneviève@gmail.com	4743044342	P14	C14
Caron	Stéphane	28/06/1956	M	71 Rue Exemple, Ville, Pays	0669864352	Caron.Stéphane@outlook.com	2340706970	P15	C15
Gauthier	Valérie	19/01/1987	F	38 Rue Exemple, Ville, Pays	0719369009	Gauthier.Valérie@gmail.com	9200329223	P16	C16
Boucher	Philippe	22/08/1988	M	58 Rue Exemple, Ville, Pays	0753058806	Boucher.Philippe@gmail.com	8026649865	P17	C17
Simard	Marie	23/07/1999	F	79 Rue Exemple, Ville, Pays	0608471535	Simard.Marie@outlook.com	9680459755	P18	C18
Leroy	David	21/12/2002	M	56 Rue Exemple, Ville, Pays	0637878830	Leroy.David@gmail.com	8408891410	P19	C19

Nom	Prenom	Specialite	Diplome	Age	sexe	Experience	Horaire	Poste	ID_Medecin_
Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre
Doe	John	Technicien de Laboratoire	Doctorat en Médecine	45	M	20	8h-17h	Chef de service	M12
Martin	Jean	Analyste Biomédical	Doctorat en Biologie Moléculaire	52	M	10	7h-16h	Technicien de Recherche	M2
Chevalier	Bruno	Chercheur en Immunologie	Master en Microbiologie	60	M	30	8h-17h	Chargé de Projet Scientifique	M3
Moreau	Pierre	Chercheur en Immunologie	Ph.D. en Immunologie	51	M	4	16h-15h	Chargé de Projet Scientifique	M4
Dubois	Marie	Biologiste Moléculaire	Doctorat en Biologie Moléculaire	52	F	32	8h-20h	Responsable de Laboratoire	M5
Girard	Paul	Analyste Biomédical	Doctorat en Biologie Moléculaire	42	M	4	8h-20h	Technicien de Recherche	M6
Lambert	Catherine	Analyste Biomédical	Master en Microbiologie	49	F	30	11h-15h	Responsable de Laboratoire	M7
Roux	François	Analyste Biomédical	Ph.D. en Immunologie	36	M	1	12h-12h	Technicien de Recherche	M8
Fournier	Claire	Biologiste Moléculaire	Ph.D. en Immunologie	69	F	2	8h-20h	Responsable de Laboratoire	M9
Legrand	Christine	Chercheur en Immunologie	Master en Microbiologie	47	F	26	16h-19h	Technicien de Recherche	M11

VI. Epreuve & Difficulté

Comme nous l'avons dit plusieurs fois précédemment, nous avons subi de nombreuses difficultés tout au long de ce projet. Ces problèmes, allons de la compréhension de logiciel ou à des problèmes plus importants dans la structure de votre code.

- Pour commencer la première difficulté est apparue lors de la création du MCD. En effet nous avons rencontré des difficultés concernant la compréhension du logiciel qui était entièrement nouveau ou encore de l'établissement des Différentes entités sur le plan théorique de la base de données, afin d'avoir La meilleure compréhension possible de l'énoncer.
- Ensuite, lors de la réalisation de ce dernier, nous en avons rencontré lors de la remise en cause de notre MCD, Car il a été très difficile. Tout en gardant un point de vue cohérent sur le laboratoire d'analyse que nous avons voulu établir au travers de la base de données.
- Pour continuer, une fois la conversion MDL SQL terminée, Nous avons eu des difficultés concernant le code en SQL. En effet, le logiciel n'étant pas parfait, nous avons dû recoder nous-mêmes, le SQL n'étant pas un langage que nous ne connaissions particulièrement à ce moment-là. Comprendre les erreurs émises par le logiciel A été une difficulté lors de ce projet.
- De plus, quand nous avons continué ce projet et sommes arrivés à API, donc l'ensemble des fonctions en code Python où nous avons utilisé fast API, une bibliothèque que nous ne connaissions pas du tout alors, malgré des difficultés Lors des premières définitions que nous avons codées. Elles sont très rapidement devenues plus faciles face à l'apprentissage de cette Librairie.

- Et enfin, lors de l'établissement de l'interface graphique, l'apprentissage de 3 nouveaux langages, complètement inconnus jusque-là, a été une réelle difficulté dans la compréhension ou encore dans l'analyse dans le développement de cette interface, ce qui a rendu les choses d'autant plus intéressantes. De plus, nous avons eu des problèmes tels que le CORS (un mécanisme qui consiste à transmettre des entêtes HTTP qui déterminent s'il faut ou non bloquer les requêtes à des ressources restreintes sur une page web qui se trouve sur un domaine externe) qui venait bloquer l'ensemble des transactions effectuées par l'API entre la base de données

et l'interface HTML cela avait pour effet de complètement bloqué l'intégralité Des tests que nous avons pu effectuer sur ces fonctions car impossible de vérifier une fonction si on ne peut même pas l'effectuer.

VII. Conclusion

Pour conclure au travers de ce projet, nous avons conçu et développé une base de données pour un laboratoire d'analyse médicale. En effet, nous avons créé une base de données afin de répondre aux différentes problématiques que le. Laboratoire d'analyse médicale faisait face. Telle que la gestion de rendez-vous, la gestion des dossiers patients ou encore la gestion de la, de la facturation Et la centralisation des informations.

Dans ce projet, malgré de nombreuses difficultés, grâce à de nouveaux outils tels que DB main et DB browser. Réussi à créer. Une base de données en SQL et l'API permettant de relier l'interface web à la base de données. Mais dans un futur proche, et si nous avions plus de temps et plus de moyens, nous aurions pu imaginer d'autres solutions pour ce projet. La première et la plus importante dans la réalisation d'une base de données pour un laboratoire, nous paraît être celle de l'hébergement afin de créer un véritable site internet permettant d'interagir à distance de n'importe où a cette base de données afin de pouvoir créer des personnes, des rendez-vous ou même des analyses.

Un autre point auquel on peut penser et le système de Login Password. En effet, un système de comptes pourrait énormément apporter de nouvelles fonctionnalités, que ce soit comme une prise de rendez-vous personnalisé, une messagerie Personnalisée ou encore un agenda ne contenant que vos rendez-vous à venir. Ou plus simplement d'accéder à l'ensemble de ces analyses en ligne.

Dans la continuité de cette idée, on peut imaginer des statistiques Comme leur médecin favori ou encore leurs horaires favoris permettant à chaque personne afin de rendre leur expérience au laboratoire cette fois-ci plus facile et plus pratique.

On peut ensuite penser à l'implémentation d'un interface HTML bien plus pratique, bien plus facile d'utilisation et bien plus optimisée Afin de rendre l'expérience utilisateur encore meilleure Et de permettre à tous et à toutes de comprendre

comment prendre un rendez-vous. Et enfin, on peut imaginer la création de meilleures animations et d'une interface web plus vivante via du CSS Afin, toujours dans un même but, de rendre l'utilisation utilisateur encore meilleur.