Un Peuple – Un But – Une Foi

REPUBLIQUE DU SENEGAL

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR DE LA RECHERCHE ET DE L'INNOVATION

UNIVERSITE GASTON BERGER DE SAINT-LOUIS

U.F.R DE SCIENCES APPLIQUEES ET DE TECHNOLOGIE

CENTRE DE FORMATION PROFESSIONNELLE PLURIDISCIPLINAIRE

RAPPORT DE STAGE DE FIN D'ETUDES POUR L'OBTENTION DU

MASTER PROFESSIONEL DEVELOPPEMENT DE SYSTEMES D'INFORMATION

Sujet : Système Électronique de Notification des Cas de VIH/SIDA au Sénégal

Présenté le 11/12/2019 par :

MOSNAN SANENGAR

Sous la direction de :

Dr. Cherif DIALLO

Maître de stage :

M. Amadou Ndiaye MBAYE

Jury :

Président :	M. David Célestin FAYE
Examinateur :	Cyrille Angelier LEPLAN
Encadrant :	M. Chérif DIALLO

Année académique 2018/2019

Remerciements

Nous ne saurions qualifier notre ingratitude si nous ne remercions tous ceux qui d'une manière ou d'une autre, ont contribué à notre éducation et à l'élaboration de ce document.

Ainsi, nous remercions tout en premier notre Dieu, le Tout Miséricordieux et lui rendons grâce pour sa bénédiction.

Nous tenons à exprimer notre profonde gratitude et nos sincères remerciements à :

- *Toute notre famille ;*
- *La Direction du CCOS et la Division des Systèmes d'Information pour nous avoir accepté au sein de l'institut pour un séjour qui a rendu possible ce travail ;*
- ***M. Amadou Ndiaye MBAYE** et **Mme Ndella DIOP**, mes maîtres de stage pour leurs orientations, encouragements et leurs disponibilités ;*
- ***Pr. Chérif DIALLO**, mon encadrant pédagogique pour sa disponibilité, son écoute et pour son encadrement ;*
- ***Dr. Babacar DIOP** pour ses conseils ainsi que le reste du corps professoral de l'UFR SAT.*

Nous n'oublions pas le reste du personnel de la division pour l'expérience enrichissante et pleine d'intérêt qu'il nous a fait vivre.

Merci à tous !

Dédicaces

A mes très chers parents.

Pour m'avoir initié très tôt sur le chemin de l'école. J'espère que vous trouvez dans ce travail un sincère témoignage de mon profond amour et éternelle reconnaissance.

Sigles et Abréviations

SIGLES	DEFINITIONS
ARV	Antirétroviraux
ASP	Active Server Pages
CCOS	Centre de Calcul Ousmane Seck
CD4	Cluster de Différenciation 4
DOM	Document Object Model
HTTP	HyperText Transfert Protocol
IIS	Internet Information Services
IRESSEF	Institut de Recherche en Santé, de Surveillance épidémiologique et de Formation
MVC	Modèle Vue Contrôleur
PVVIH	Personne Vivant avec le VIH
SGBD	Système de Gestion de Base de Données
SIDA	Syndrome Immunodéficience Acquise
TDR	Test de Diagnostic Rapide
UML	Unified Modeling Language
VIH	Virus de l'Immunodéficience Humaine

Résumé

Le présent document est le rapport du stage effectué pour valider le Master en Développement de Systèmes d'Information du Centre de Formation Professionnelle Pluridisciplinaire (CFPP).

En effet, il est recommandé aux étudiants en fin de formation de rédiger un mémoire ou d'effectuer un stage en entreprise afin de découvrir les réalités pratiques en entreprise. Dans le cas où l'étudiant choisi la deuxième option, il lui faut rédiger un rapport pour rendre compte du travail accompli au cours de cette période.

Le Centre de Calcul Ousmane Seck (CCOS) de l'Université Gaston Berger nous a servi de cadre pour le nôtre.

L'Université Gaston Berger, dans le cadre de sa mission de service à la communauté, a mis en place une plateforme de notification des cas de VIH au Sénégal pour le compte de l'Institut de Recherche en Santé, de Surveillance Epidémiologique et de Formation (IRESSEF).

Le Système Electronique de Notification des Cas de VIH/SIDA au Sénégal (SENCAS) est en production depuis 2012 et fréquemment de nouveaux besoins sont exprimés. Notre projet de fin d'études a consisté en une refonte de cette application et un rajout des fonctionnalités spécifiques à Neema. Les objectifs des travaux sont principalement d'adopter une architecture logicielle qui permettrait de prendre en compte les évolutions de l'application tout en évitant une complexification progressive du système et de faciliter sa prise en main par une autre équipe de développeurs. En résumé, le but est d'assurer une meilleure efficacité lors des activités de maintenance évolutive et quelques fois corrective. Seul l'applicatif est concerné par cette refonte, la base de données étant conservée.

Ce document présente les différentes étapes effectuées pour la réalisation de ce projet.

En effet, l'application est basée sur une architecture MVC de la technologie ASP.NET Core et tourne sur le SGBD SQL Server.

Mots Clés : SQL Server, Asp.Net MVC ,Web Forms, Framework .Net, .Net Core

Abstract

This document is the report of the internship carried out to validate the Master in Development of Information Systems of the Center for Multidisciplinary Professional Training (CFPP).

Indeed, it is recommended to students at the end of training to write a dissertation or to do an internship in a company to discover the practical realities in business. In the case where the student chooses the second option, he must write a report to report on the work accomplished during this period.

The Ousmane Seck Computing Center (CCOS) at Gaston Berger University provided us with a framework for ours.

Gaston Berger University, as part of its community service mission, has set up an HIV case reporting platform in Senegal on behalf of the Institute for Health Research, Epidemiological Surveillance and Training (IRESSEF).

The Electronic HIV/AIDS case Notification System in Senegal (SENCAS) has been in production since 2012 and frequently new needs are expressed. Our graduation project consisted of a redesign of this application. The objectives of the work are mainly to adopt a software architecture that would take into account the changes in the application while avoiding a progressive complexity of the system and facilitate its handling by another team of developers. In summary, the goal is to ensure better efficiency during evolutionary and sometimes corrective maintenance activities. Only the application is concerned by this redesign, the database being retained.

This document presents the various steps taken to complete this project.

Indeed, the application is based on an MVC architecture of ASP.NET Core technology and runs on the SQL Server DBMS.

Keywords: SQL Server, Asp.Net MVC ,Web Forms, Framework .Net, .Net Core

Liste des Figures et des tableaux

Figure 1 : Organigramme du CCOS 5
Figure 2:Page Webform(w3c.org) 9

Figure 3:Architecture MVC(w3schools) 10
Figure 4:Arch. ASP.NET Core(docs.microsoft.com) 16
Figure 5:Diagramme des cas d'utilisation 20
Figure 6:Diagramme de classe 22
Figure 7:Diagramme de classe 23
Figure 8:Diagramme de classe 23
Figure 9:Diagramme de classe 24
Figure 10:Diagramme de classe 24
Figure 11:Diagramme de classe 25
Figure 12:Diagramme de classe 26
Figure 13:Diagramme de classe 26
Figure 14:Diagramme de classe 27
Figure 15:Diagramme de classe 27
Figure 16: Diagramme de classes ajoutées 30
Figure 17:page d'authentification de la plateforme 33
Figure 18:Tableau de bord de l'administrateur du système 34
Figure 19:Formulaire de saisie 34
Figure 20:Liste des patients 35
Figure 21:Outils de génération 38
Figure 22:Contenu du Modèle 39
Figure 23:Choix de la connexion 40
Figure 24:Choix de la base 41
Figure 25:Test de connexion 42
Figure 26:Choix des tables, des vues et des procédures stockées 43

Table des matières

Remerciements ii

Dédicaces iii
Sigles et Abréviations iv
Résumé v
Abstract vi
Liste des Figures et des tableaux vi
Introduction 1
CHAPITRE 1 : Présentation du cadre du stage 3
1.1 Centre de Calcul Ousmane Seck 3
1.1.1 Historique 3
1.1.2 Missions 4
1.1.3 Structure organisationnelle 4
1.2 L'équipe 5
1.3 Contexte d'étude 5
1.3.1 Projet 5
1.3.2 Objectifs 6
CHAPITRE 2 : Etat de l'art et Technologies 7
2.1 Introduction 7
2.2 ASP.NET 7
2.2.1 ASP.NET WEB FORMS 7
2.2.2 Système de gestion de base données 9
2.2.3 ASP.NET MVC 10
2.2.4 Différences entre ASP.NET MVC et Web Forms 11
2.3 Choix de la technologie 12
2.4 ASP .NET MVC Core 13
2.4.1 Entity Framework Core 14
2.4.2 Serveur IIS 15
2.5 Environnement de développement 16
CHAPITRE 3 : Spécifications et Réalisation du projet 17
3.1 Spécifications fonctionnelles 17
3.2 Diagramme de cas d'utilisation 18
3.3 Diagramme de classes 21
3.4 Outils et technologies utilisés 30
3.4.1 Technologies 30
3.4.2 Outils Microsoft Visual Studio 32

3.5 Implémentation de SENCAS 32
3.5.1 Saisie 34
3.5.2 Liste 35
Conclusion 36
Bibliographie et webographie 37
Références bibliographiques 37
Références webographiques 37
Annexes 38

Introduction

Au Sénégal, les personnes diagnostiquées positives à l'infection au VIH/SIDA ont un suivi régulier dans les structures de santé. Lors du dépistage, tous les patients sont enregistrés grâce à un numéro d'anonymat et des informations personnelles sont recueillies manuellement à l'aide d'outils de collecte (registres, fiches de dépistage…).

Au niveau clinique, il existe des outils pour la collecte des données et le suivi des personnes infectées (caractéristiques sociodémographiques, diagnostic biologique et TDR, comptage CD4 et charge virale, stade clinique et survenue d'infection opportunistes, traitement ARV…).

Malgré leurs mises en place, ces outils de collecte ne permettent pas la remontée des données individuelles des personnes vivant avec le VIH (PVVIH), c'est ainsi qu'un outil électronique de renforcement du système de surveillance épidémiologique du VIH au Sénégal dénommé **Système Electronique de Notification des Cas VIH/SIDA au Sénégal** a été réalisé en 2012 grâce à la collaboration des informaticiens du Centre de Calcul de l'UGB de Saint-Louis. Cette nouvelle forme électronique de notification permet de recueillir de façon continue, d'analyser et d'interpréter des données des personnes vivant avec le VIH et de communiquer à temps opportun des informations ainsi obtenues aux décideurs et aux autres instances concernées.

Cependant, l'application étant développée dans une architecture vieillissante non cohérente, l'intégration continue des nouvelles fonctionnalités à la demande du partenaire devient de plus en plus difficile. De plus, la duplication des codes sources et bien d'autres problèmes techniques apparaissent et rendent la maintenance de la plateforme fastidieuse.

Face aux inquiétudes, de nombreuses interrogations ont été émises par l'équipe conceptrice du système :

Quelles stratégies adoptées pour prendre en compte les évolutions futures de l'application en fonction du besoin métier ?

Comment éviter une complexification progressive de l'application dans le temps pour faciliter sa prise en main, c'est-à-dire rendre possible l'intervention d'une personne ne connaissant pas le projet ?

C’est dans ce contexte, qu’il nous a été confié, durant notre stage de refondre l’existant tout en prenant en charge des recommandations et fonctionnalités supplémentaires dans une architecture moderne. En effet, celle-ci est destinée à apporter des solutions pour préserver sa maintenance, ou mieux, améliorer son évolutivité et sa simplicité afin de garantir sa stabilité et sa pérennité.

Nous commencerons par la présentation de notre cadre de stage (chapitre 1) pour parler dans un second temps de l’architecture et des plateformes de l’existant ainsi que des technologies retenues pour le développement du projet (chapitre 2). Nous continuerons ensuite par les différents approfondissements effectués, notamment les fonctionnalités supplémentaires prises en charge dans la nouvelle version (chapitre 3). Finalement nous conclurons, dresserons un bilan de ce qui a été réalisé et présenterons les perspectives d’avenir du projet.

CHAPITRE 1 : *Présentation du cadre du stage*

Dans ce premier chapitre, nous présentons d'abord la structure d'accueil CENTRE DE CALCUL OUSMANE SECK, l'équipe dans laquelle nous avons travaillé avant de terminer par le contexte d'étude.

1.1 Centre de Calcul Ousmane Seck

1.1.1 Historique

Dès l'ouverture de l'Université de Saint-Louis en décembre 1990, une des priorités des autorités de la nouvelle Université était de dispenser des cours d'informatique à tous les étudiants dès la première année d'inscription. C'est ainsi que la nécessité de mettre en place une structure pouvant répondre à ce souci de généralisation d'un enseignement jusqu'alors réservé a conduit à l'érection d'un Centre de Calcul jouissant du label " service commun ".

Sa mission première était d'abriter tous les cours d'informatique des quatre U.E.R (Unité d'Enseignement et de Recherche) devenues plus tard U.F.R. (Unité de Formation et de Recherche).

L'U.F. R des Sciences Appliquées et de Technologie qui, comme son nom l'indique, avait une vocation de former des informaticiens, s'est retrouvée tout naturellement gestionnaire du Centre de Calcul.

Le Centre de Calcul a été baptisé en 1994, à la mémoire de "Ousmane SECK" cet éminent enseignant en Mathématiques et Informatique de l'ESP (ex ENSUT) qui a consacré une bonne partie de ses enseignements à notre Institution.

Il décéda le 9 décembre 1993, des suites d'un accident d'avion qui le ramenait de Saint-Louis à Dakar après une journée très chargée à l'université de Saint Louis.

En 1996, le séminaire d'évaluation a été l'occasion pour l'ensemble de la communauté universitaire de réfléchir sur tous les aspects d'un développement fructueux et durable de notre Institution. L'idée avait été retenue d'ériger le Centre de Calcul en Institut d'Université.

Il faudra attendre 1999, pour que l'Assemblée de l'Université, en sa séance du 21 avril, délibère sur cette question en émettant un avis favorable sur le projet de décret qui lui était soumis faisant du Centre de Calcul un Institut d'Université.

En attendant la signature du projet de décret, le Recteur de l'Université Gaston Berger, par arrêté **n° 031 du 31 janvier 2000**, a mis à la tête de cette structure un Coordonnateur chargé de mettre en place, progressivement, un organigramme de son administration et d'élaborer le projet d'une nouvelle orientation.

1.1.2 Missions

Compte tenu du développement des Technologies de l'Information et de la Communication, le Centre de Calcul, chargé du programme d'informatisation de l'Université et avec les missions d'enseignement, de recherche et de formation qui lui sont dévolues (décret **2003-594** du 16 juillet 2003 fixant les règles d'organisation et de fonctionnement du Centre de Calcul de l'Université Gaston Berger de Saint-Louis), se trouve aujourd'hui au cœur du système de formation de l'Université.

L'une des ambitions du Centre de Calcul est de contribuer à un plus grand accès de l'université et des établissements d'enseignement et de recherche de la région aux NTIC et de leur permettre de participer efficacement au transfert et à l'échange des connaissances scientifiques et techniques par l'information et la communication.

1.1.3 Structure organisationnelle

Pour la réalisation de sa mission, le Centre de Calcul Ousmane s'appuie sur trois services :

- ✓ Les services Administratifs :
- ✓ Le Service des Infrastructures et des Systèmes d'Information :
- ✓ Le Service Enseignement, Formation et Recherche :

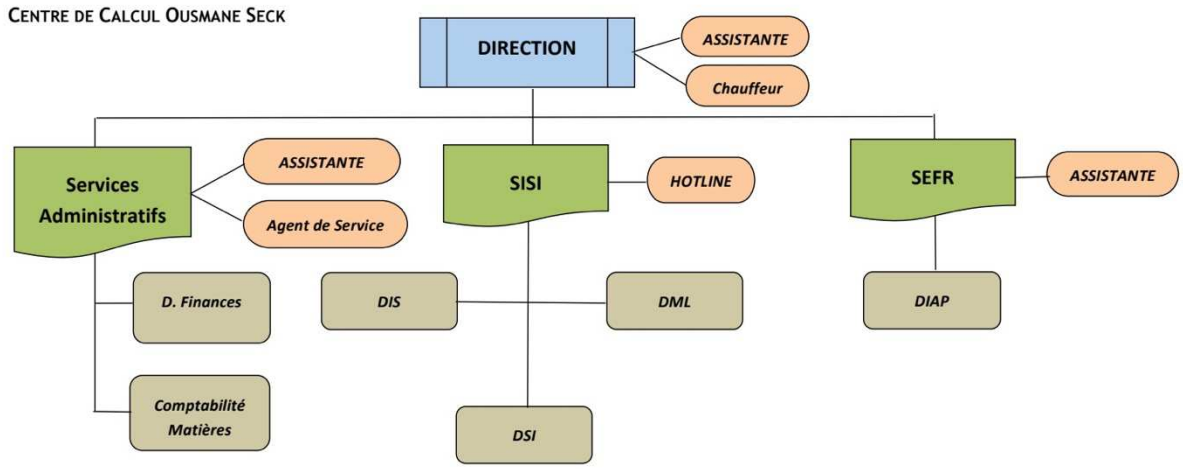


Figure 1 : Organigramme du CCOS

1.2 L'équipe

Durant ce stage, nous avons intégré la Division des Systèmes d'Information.

Dans cette division, l'équipe compte actuellement sept (07) personnes dont quatre (04) ingénieurs et trois (03) stagiaires organisés selon les différents projets. Cette division répond essentiellement :

- Aux problématiques liées aux systèmes d'information de l'Université ;
- Aux problématiques « métier » de ses partenaires via des solutions décisionnelles.

1.3 Contexte d'étude

1.3.1 Projet

Notre stage s'inscrit dans un projet de refonte d'une application Web dans une architecture moderne. Cette refonte a été formulée par l'équipe conceptrice du projet. Elle intervient dans le cadre de rajout des parties Neema. Ces parties sont des fonctionnalités supplémentaires sur une demande de Intrahealth dans le cadre du projet USAID Neema dont l'un des objectifs est la meilleure maitrise de l'évolution de l'infection du VIH dans les populations clés. C'est suite à cette demande que l'équipe a décidé de refondre la solution existante pour améliorer l'aspect technique de l'application.

1.3.2 Objectifs

L'objectif de ce projet est de changer l'architecture et les technologies de la solution existante pour apporter des solutions à la maintenance évolutive et faciliter l'intervention des nouveaux développeurs dans le projet.

Bien que sa maintenance soit fastidieuse, la solution existante reste tout de même en production avec trois acteurs. Elle se limite seulement aux personnes dépistées positives au VIH.

Un document des spécifications expliquant l'expression des besoins concernant cette nouvelle application nous a été fourni pour mener à bien notre travail.

Celui-ci contient :

- ✓ La description de la solution existante ;
- ✓ Ajout des nouveaux acteurs ;
- ✓ Prises en comptes des cas négatifs ;
- ✓ Ajout des quartiers pour la recherche des communes de naissance ;
- ✓ Spécifications du formulaire de l'acteur Neema
- ✓ Etc.

Le choix du langage des technologies permettant d'implémenter l'application nous a été confié. La seule contrainte imposée est l'architecture MVC.

Le développement du projet se déroule sous la supervision de notre maître de stage.

Nous aborderons l'état de l'art et les technologies concernés dans le chapitre suivant.

CHAPITRE 2 : *Etat de l'art et Technologies*

2.1 Introduction

Dans le cadre de la refonte d'une application web de notification des cas de VIH au Sénégal, la Division Systèmes d'Information du CCOS nous a missionné pendant notre stage pour le développement de la version 2.0 autour des technologies Microsoft ASP.NET MVC Core avec une gestion des données sous le moteur SGBD SQL Server.

A cet effet, il est nécessaire d'analyser les technologies et les plateformes utilisées pour la conception et la réalisation de l'ancienne application afin de répondre efficacement à la problématique posée.

Dans cette partie, nous mettons l'accent sur les plateformes et les technologies utilisées pour la réalisation de l'existant qui sont notamment ASP.NET Web Forms et SQL Server de Microsoft. Nous finirons par celles qui sont retenues pour le développement de la version 2.0.

2.2 ASP.NET

ASP.NET est une plate-forme de développement Web qui fournit un modèle de programmation, une infrastructure logicielle complète et divers services nécessaires à la création d'application robuste pour PC ainsi que pour les appareils mobiles.

Il s'agit d'une évolution majeure d'Active Server Pages[1] (ASP classique créé en 1996), par laquelle cette technique a été incorporée au Framework .NET créé aussi par Microsoft, d'où le nom .Net après ASP. Ces applications peuvent être développées à l'aide de langages de programmation tels que C#, Visual Basic, Java Script, F# ... De plus, divers styles de développement sont disponibles dans ASP.NET pour créer des applications.

Web Forms et MVC sont deux d'entre eux.

Dès sa sortie, ASP.NET utilise le moteur Web Forms comme moteur de rendu (les pages .aspx et les fameux tags <% et %>).

2.2.1 ASP.NET WEB FORMS

[1] Active Server Pages (ASP) : Cadre de développement pour la création de pages web.
ASP est une structure composée d'objet accessibles par deux langages principaux : VBScript et JScript.

Web Forms est le modèle de programmation ASP.NET le plus ancien avec des pages web pilotées par les évènements, combinant HTML, les contrôles serveur et le code serveur. Il suit un modèle de développement traditionnel basé sur les événements.

Le navigateur envoie un formulaire Web au serveur, qui renvoie une page de balisage complet ou une page HTML en réponse. Toutes les activités utilisateur coté client sont transférées au serveur pour traitement avec état. Le serveur traite la sortie des actions du client et déclenche les réactions. Il est compatible avec tous les navigateurs et langages pris en charge par .Net Common Langage Runtime[2] (CLR). Il est inclus dans Visual Studio.

Contrairement à ASP classique ou il est impossible de séparer le code exécutable du code HTML, Web Forms résout le problème avec les contrôles serveur qui ne sont que des balises comprises par le serveur.
Il existe trois types de contrôles serveur :

- ✓ Contrôles du serveur HTML : Ce sont des balises HTML traditionnelles dans lesquelles figure l'attribut runat pour indiquer que l'élément doit être traité sur le serveur ;
- ✓ Contrôles de serveur Web : ce sont des balises ASP.NET spéciales créées sur le serveur qui nécessite également l'attribut runat pour fonctionner ;
- ✓ Commandes de serveur de validation : ils servent à valider les entrées de l'utilisateur c'est-à-dire si la saisie de l'utilisateur n'aboutit pas, un message d'erreur lui sera envoyé.

Ce modèle a pour avantage de fournir une vaste palette de contrôle permettant le prototypage rapide d'une application. Cependant, il n'est pas possible de contrôler directement le code HTML généré en sortie.

2.2.1.a Exemple de code Web Forms

[2] **Common Language Runtime** (CLR) : Implémentation de la norme CLI par Microsoft
Permettant la compilation du code source de l'application en un langage intermédiaire

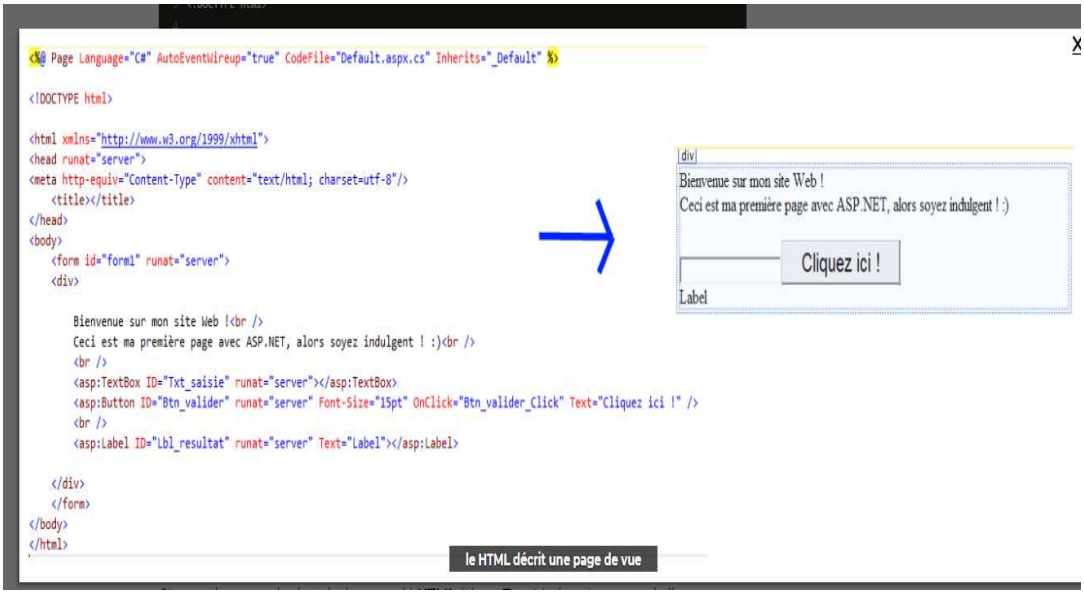


Figure 2:Page Webform(w3c.org)

2.2.1.b Framework .Net

Le Framework .Net est un Framework créé par Microsoft en 2002. Il est principalement dédié à la réalisation d'applications s'exécutant dans des environnement Microsoft. Il offre la possibilité de réaliser des programmes fonctionnant sous Windows ou des sites web ou encore des applications pour appareils mobiles.

Il a été conçu dans le but de faciliter le travail des développeurs en unifiant l'approche de conception d'application Windows.

L'environnement d'exécution du Framework se compose d'un moteur d'exécution, appelé Common Language Runtime (CLR). Il s'agit d'une implémentation de la norme CLI par Microsoft permettant la compilation du code source de l'application en un langage intermédiaire, nommé Microsoft Intermédiate Language et se comporte comme la machine virtuelle de Java. Ainsi tous les langages compatibles respectant la norme CLI ont accès à toutes les bibliothèques disponibles dans l'environnement d'exécution. Le Framework .Net est un sous ensemble de la technologie Microsoft .Net.

2.2.2 Système de gestion de base données

La solution développée utilise comme système de gestion de base données (SGBD) Microsoft SQL Server. Comme tout autre (SGBD), il permet le stockage et la gestion des données. Il a pour particularité d'être un SGBD supportant nativement des requêtes SQL qui implique plusieurs bases de données.

Microsoft SQL Server utilise le langage T-SQL(Transact-SQL) pour ses requêtes. Il s'agit d'une implémentation de SQL prenant en charge les procédures stockées[3] et les déclencheurs[4]. Il possède également une composante transactionnelle qui lui permet de préparer des modifications sur les données d'une base de les valider ou les annuler de façon automatique. Cela permet de garantir la cohérence et l'intégrité des données stockées dans la base de données.

2.2.3 ASP.NET MVC

ASP.NET MVC est essentiellement une infrastructure de développement Web de Microsoft qui combine les fonctionnalités de l'architecture MVC (Modèle-Vue-Contrôleur), les idées et les techniques les plus récentes du développement agile et les meilleures parties de l'ASP existant. C'est une alternative complète et plus légère au Web Forms traditionnel. Annoncé en 2007, il a été publié en Avril 2009.

En Mars 2012, Microsoft avait publié une partie de sa pile Web (y compris ASP.NET MVC, Razor et Web API) sous une licence open source (apache Licence 2.0).

La figure ci-après illustre l'architecture MVC qui est un patron de conception (design pattern) développé en 1970, qui sert à séparer les applications en trois couches :

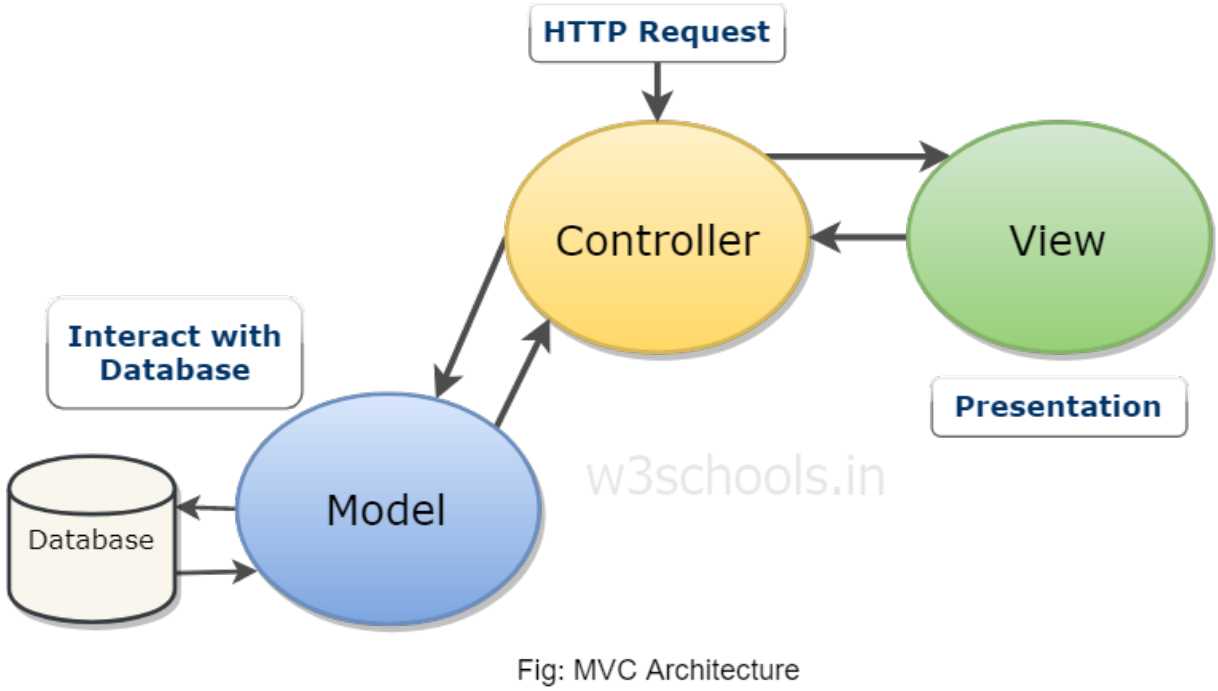


Figure 3:Architecture MVC(w3schools)

[3] **Procédures stockées** : Ensemble d'instructions SQL précompilées, stockées dans une base de données et exécutées sur demande par le SGBD qui manipule la base.
[4] **Déclencheurs** : Objet logiciel qui provoque un traitement particulier en fonction d'événements prédéfinis

Trois couches sont représentées :

- ✓ Modèle (Model) : décrit les données manipulées par l'application telles que les interactions avec la base de données et est responsable de leurs intégrités.
- ✓ Vue (View) : représente la partie de l'application qui dirige l'affichage des données récupérées depuis le modèle ;
- ✓ Contrôleur (Controller) : est la partie qui s'occupe de la gestion des interactions de l'utilisateur avec l'application. En général, le contrôleur lit les données affichées depuis la vue, récupère les interactions de l'utilisateur et envoie l'ensemble des données au modèle ;

L'avantage principale de ce modèle est la séparation de la vue, du modèle et du contrôleur rendant le code plus facile à maintenir. De plus, contrairement à Web Forms, le développeur a un contrôle total sur le code HTML rendu.

Voilà de façon générale la description des deux technologies de Microsoft. Voyons ce qui en est de leurs différences.

2.2.4 Différences entre ASP.NET MVC et Web Forms

Le tableau ci-dessous récapitule de manière non exhaustive les principales différences entre Web Forms et ASP.NET MVC :

Tableau 1 : tableau comparatif

ASP.NET Web Forms	ASP.NET MVC
Asp.Net Web Forms suit un modèle de développement traditionnel basé sur les événements.	Asp.Net MVC est léger et suit le modèle de développement basé sur le pattern MVC (Modèle, Vue, Contrôleur).
Asp.Net Web Forms a des contrôles serveur.	Asp.Net MVC a des helpers HTML.
Asp.Net Web Forms prend en charge l'état d'affichage pour la gestion de l'état côté client.	Asp.Net MVC ne prend pas en charge l'état d'affichage.
Asp.Net Web Forms a des URL basées sur les fichiers, ce qui signifie que le nom de fichier existant dans les URL doit avoir son existence physique.	Asp.Net MVC a des URL basées sur la route, ce qui signifie que les URL sont divisées en contrôleurs et actions. De plus, elles sont basées sur le contrôleur et non sur le fichier physique.

Asp.Net Web Forms suit la syntaxe des formulaires Web	Asp.Net MVC suit une syntaxe personnalisable (Razor par défaut)
Dans Asp.Net Web Forms, les formulaires Web (ASPX), c'est-à-dire les vues, sont étroitement couplés à la logique Code behind (ASPX.CS).	Dans Asp.Net MVC, les vues et la logique sont conservés séparément.
Asp.Net Web Forms a des pages maîtres pour une apparence cohérente.	Asp.Net MVC a des dispositions pour une apparence et une cohérence constante.
Asp.Net Web Forms possède des contrôles utilisateur pour la réutilisation du code.	Asp.Net MVC dispose de vues partielles pour la réutilisation du code.
Asp.Net Web Forms a des contrôles de données intégrés et est idéal pour le développement rapide avec un accès puissant aux données.	Asp.Net MVC est léger, permet un contrôle total du balisage et prend en charge de nombreuses fonctionnalités permettant un développement rapide et agile. Par conséquent, il est préférable de développer une application Web interactive avec les normes Web les plus récentes.
Asp.Net Web Forms n'est pas Open Source.	Asp.Net Web MVC est une source ouverte.

Il va de ce tableau que ASP.NET MVC peut répondre à notre problématique car son organisation architecturale rend le code facile à maintenir.

2.3 Choix de la technologie

Pour le choix de la technologie à utiliser pour ce travail de refonte, nous avons tout d'abord vérifié le système de gestion des données (SQL Server) sur lequel viendra se pose notre applicatif. Ce qui, à priori, nous pousse à nous dire que l'utilisation d'un langage de programmation conçu par Microsoft nous permettrait de profiter des optimisations liées à l'environnement Windows, mais aussi d'avoir moins de configurations et d'installations à effectuer pour mettre en place l'environnement de travail et ainsi commencer directement la partie programmation de l'application. C'est ainsi qu'après concertation avec mon maître de stage notre choix s'est orienté vers ASP.NET MVC.

Après sélection du modèle de programmation il restait à déterminer lequel de ASP.NET MVC (.Net Framework) et ASP.NET MVC Core (.Net Core) choisir pour non seulement répondre à la problématique posée mais aussi être à jour dans une technologie en parfaite évolution. Le second convient le mieux aux exigences et apparait pour nous comme une solution adéquate car il regroupe toutes les fonctionnalités de MVC et est multiplateforme.

2.4 ASP .NET MVC Core

ASP.NET Core est une infrastructure Web open source et optimisée pour le cloud permettant de développer des applications Web modernes et des services Web, des applications IoT et des back-ends mobiles pouvant être développées et exécutées sous Windows, Linux et Mac. Il inclut le Framework MVC, qui regroupe désormais les fonctionnalités de MVC et l'API Web dans un Framework de programmation Web unique.

Contrairement à .Net Framework qui fonctionne uniquement sur Windows ou Windows Server, .Net Core peut être utilisé aussi bien sur Windows que sur Linux ou sur OS.

Il présente les avantages suivants :

- ✓ Open source et centré sur la communauté
- ✓ Comporte un certain nombre de modifications architecturales qui se traduisent par un cadre beaucoup plus léger et modulaire ;
- ✓ ASP.NET Core est basé sur un ensemble de paquet NuGet[5] granulaires et bien factorisés ;
- ✓ Permet d'optimiser l'application pour inclure uniquement les paquets NuGet nécessaires ;
- ✓ Sécurité renforcée, maintenance réduite, performance améliorée et une réduction de cout.

ASP.NET Core apporte les améliorations suivantes :

- ✓ Créez et exécutez des applications ASP.NET multi-plateformes sous Windows, Mac et Linux.
- ✓ Construit sur .NET Core, qui prend en charge la véritable gestion des versions d'applications côte à côte.

[5] **NuGet** : Gestionnaire de packages gratuit et open source, conçu pour la plateforme de développement Microsoft. Il est fourni en tant qu'extension de Visual Studio.

- ✓ Nouvel outil simplifiant le développement moderne de Web.
- ✓ Pile Web unique alignée pour l'interface utilisateur Web et les API Web.
- ✓ Configuration basée sur l'environnement compatible avec le cloud.
- ✓ Prise en charge intégrée pour l'injection de dépendance.
- ✓ Tag Helpers qui rend le balisage Razor plus naturel avec HTML.
- ✓ Possibilité d'héberger sur IIS, Nginx, Apache, Docker ou d'hôte autonome dans votre propre processus.

ASP.NET Web Core 2 offre des modèles de projet suivant :

1. Application Web principale ASP.NET (pages de rasoir).
2. Application Web ASP.NET Core (Modèle-Vue-Contrôleur).
3. API Web ASP.NET Core (sans interface utilisateur).
4. ASP.NET Core avec Angular (SPA).
5. ASP.NET Core avec React.js (SPA).
6. ASP.NET Core avec React.js et Redux (SPA).

ASP.NET Core s'intègre parfaitement avec les Frameworks et les bibliothèques populaires côté client, notamment Blazor, Angular, React et Bootstrap, et des workflows de développement.

2.4.1 Entity Framework Core

Entity Framework Core (EF Core) est la dernière version de Entity Framework de Microsoft. Il a été conçu pour être léger, extensible et prendre en charge le développement multiplateforme dans le cadre du Framework .Net Core de Microsoft.

EF Core est un Mappeur-Objet-Relationnel (ORM) de ASP.NET Core, une technique qui permet aux développeurs de travailler avec les données de manière orienté objet en effectuant un mapping entre les objets définis dans un langage de programmation et les données stockées dans la base.

Les ORM sont des bibliothèques de code pré-écrites qui permettent de :

- ✓ Mapper un modèle de domaine sur des objets de la base de données ;

- ✓ Créer des bases de données et gérer le schéma conforment aux modifications apportées au modèle ;
- ✓ Générer du SQL et l'exécuter sur la base de données ;
- ✓ Gérer les transactions ;
- ✓ Garder une trace des objets qui ont déjà été récupérés.

Il existe l'approche EF Core Code-First qui consiste à créer d'abord les classes pour nos entités de domaine puis créer ensuite la base de données à partir de notre code en utilisant des migrations. Et l'approche Database-First, dans laquelle nous concevons d'abord la base de données pour générer ensuite les modèles.

Il est à noter que nous avons utilisé l'approche Database-First pour notre développement car l'approche Code-First n'est utile que dans le cas d'un nouveau projet ou nous ne savons pas encore à quoi notre base de données devrait ressembler.

2.4.2 Serveur IIS

Les applications ASP.NET sont gérées par Internet Information Services (IIS). Il s'agit d'un serveur Web développé par Microsoft qui fournit une plateforme sécurisée et facile à administrer pour l'hébergement des services web et des applications Web.

Il peut contenir une ou plusieurs applications Web et offre la possibilité de configurer le niveau d'isolation entre ces différentes applications. Sa fonctionnalité pool d'application offre la possibilité de :

- ✓ Grouper des sites et applications qui s'exécutent avec les mêmes paramètres de configuration ;
- ✓ Isoler les sites et applications Web qui s'exécutent avec des paramètres de configuration unique ;
- ✓ Prévenir que les ressources d'une application accèdent aux ressources d'une autre.

L'architecture générale de ASP.NET Core est représentée sur la figure ci-dessous :

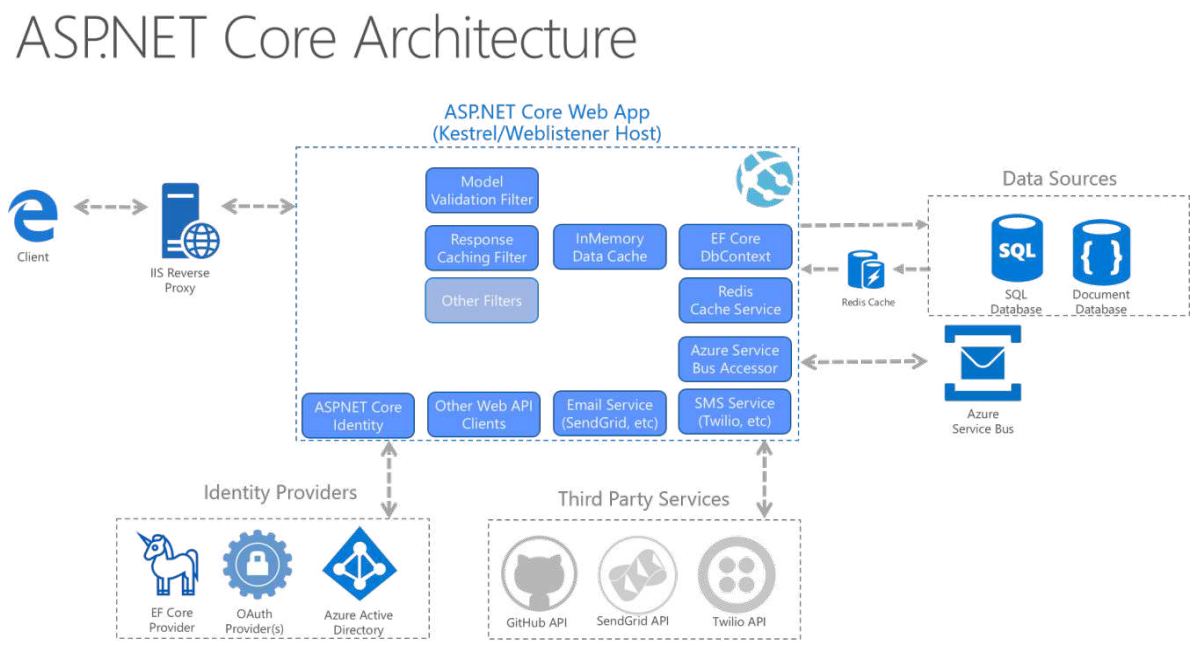


Figure 4:Arch. ASP.NET Core(docs.microsoft.com)

2.5 Environnement de développement

Choisissant la technologie ASP.NET Core, l'outil le plus adéquat pour réaliser l'application et ses différentes fonctionnalités, est l'environnement de développement Visual Studio 2019. L'utilisation des Framework Bootstrap et JS nous a permis de mettre en place les différents formulaires présents dans l'application.

Microsoft Visual Studio est environnement de développement intégré (IDE) conçu par Microsoft.

CHAPITRE 3 : *Spécifications et Réalisation du projet*

Cette partie porte sur l'analyse des besoins des utilisateurs en termes de fonctionnalités du système. Nous utiliserons l'outil d'analyse de Microsoft Visual studio 2012 pour modéliser nos différents profils utilisateurs. Ensuite nous présenterons quelques captures du diagramme de classes que nous avons généré toujours avec Visual Studio 2012 à partir de la base de données existante et dont nous nous conformons pour le développement de notre applicatif.

3.1 Spécifications fonctionnelles

Cette nouvelle version prendra totalement tous les objectifs initiaux de l'ancienne version avec évidement d'autres spécifications supplémentaires qui sont :

- ✓ Ajout de deux (02) nouveaux profils utilisateurs (Editeur-Neema et Statisticien-Neema) ;
- ✓ Prise en compte des cas négatifs ;
- ✓ Prise en compte du type de population auquel appartient le patient ;
- ✓ Identification du niveau de prestation ;
- ✓ Etc.

L'application permettra :

- ✓ D'obtenir des informations sur l'ensemble des personnes dépistées de façon générale puis par type de sérologie en particulier ;
- ✓ Suivre l'évolution du nombre de cas de séropositifs diagnostiqués ;
- ✓ Suivre la dynamique de l'infection et évaluer au plus près l'impact des programmes afin d'ajuster les actions de prévention ;
- ✓ Analyser les données et caractériser l'infection à VIH/Sida dans une catégorie de population ;
- ✓ Etc.

Afin d'assurer l'anonymat pour garantir la confidentialité et prévenir les doublons, les patients sont identifiés par un code unique. Aucune information personnelle permettant l'identification n'est stockée. Tous les fichiers électroniques et les programmes en réseau ne se feront qu'avec

le code du patient. L'accès à la plateforme sera protégé par un système d'authentification et par niveau d'accréditation.

3.2 Diagramme de cas d'utilisation

En UML, les diagrammes des cas d'utilisation sont utilisés pour donner une vision globale du comportement fonctionnel du système logiciel. Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés des acteurs.

Les acteurs sont des entités externes qui interagissent avec le système.

SENCAS fait intervenir les acteurs suivants :

- ✓ **Administrateur :** il a la charge de la gestion des comptes utilisateurs et de toutes les autres fonctionnalités de l'application. Il doit toutefois s'authentifier au préalable via un login et un mot de passe qui lui sont fourni par défaut.
- ✓ **Editeur Sencas :** il est chargé des saisies, des modifications, de rechercher et visualiser la liste des patients de sa structure.
- ✓ **Statisticien Sencas** : Il est chargé des traitements statistiques spécifiques des informations sur les patients enregistrés par l'éditeur Sencas que nous appellerons patients Sencas.
- ✓ **Editeur Neema :** il est en charge des ajouts, des modifications, rechercher et visualiser la liste des patients de sa structure et d'autres informations spécifiques au profil Neema que nous appellerons patients Neema.
- ✓ **Statisticien Neema :** il est chargé de faire des analyses statistiques des données des patients enregistrés par l'éditeur Neema.

Toutes ces actions ne peuvent être menées qu'avec au préalable l'authentification imposée à tous. La consultation de la liste des patients est restreinte aux patients de la structure de l'utilisateur connecté. Cette restriction ne concerne que les acteurs Editeur Sencas et Editeur Neema. Quant aux statisticiens, ils sont habiletés à consulter la liste complète de tous les patients de leurs éditeurs respectifs.

Le tableau suivant présente le récapitulatif des cas d'utilisation de notre système.

Tableau 2 : Cas d'utilisation

Cas d'utilisation		Acteur
S'authentifier		Tous les acteurs

Ajouter Modifier Supprimer Rechercher Consulter liste Gestion paramètre	Utilisateur Patient	Administrateur
Ajouter Modifier Rechercher Consulter liste	Patient Sencas/Structure	Editeur Sencas Statisticien Sencas
Consulter liste Rechercher Analyser données	Patients Sencas	Statisticien Sencas
Ajouter Modifie Rechercher Consulter liste	Patient Neema/Structure	Editeur Neema Statisticien Neema
Consulter liste Rechercher Analyser données	Patients Neema	Statisticien Neema

Au vu des différents profils utilisateurs et des cas d'utilisations ci-dessus, il ressort un diagramme global des acteurs suivant :

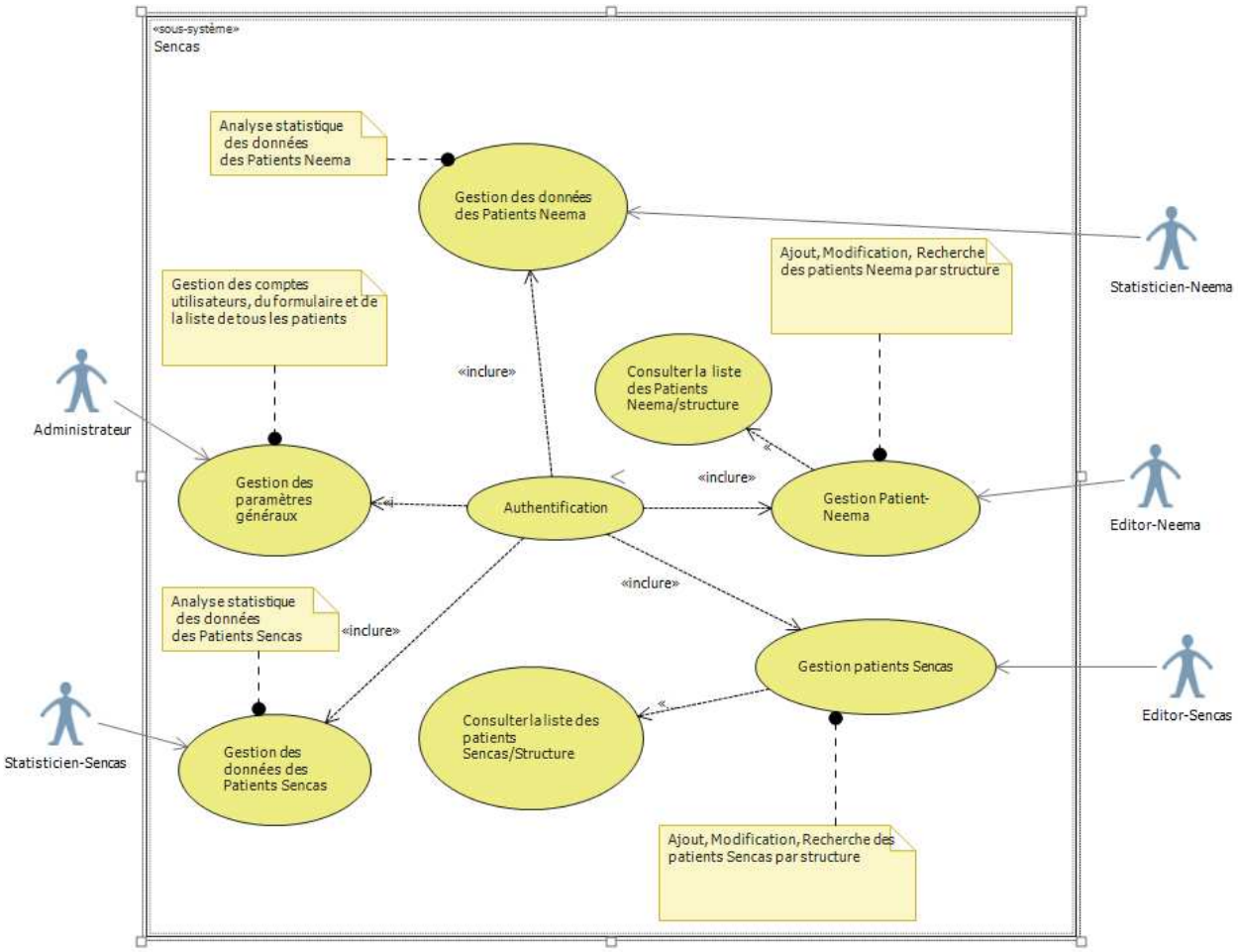


Figure 5:Diagramme des cas d'utilisation

Dans la suite nous donnerons en détail une description textuelle de quelques cas d'utilisation de notre système.

- ✓ **S'authentifier :** il s'agit de vérifier la conformité des informations de connexion (Login et mot de passe). Le système redirige l'utilisateur vers son profil si les informations entrées sont correctes. Sinon le système affiche le message d'erreur indiquant que le login ou le mot de passe est incorrect. Après cinq tentatives de connexion, le compte de l'utilisateur concerné sera bloqué.
- ✓ **Gestion des comptes utilisateurs :** permet à l'administrateur d'ajouter, de modifier, de supprimer ou de rechercher un utilisateur.
- ✓ **Gestion des patients Neema :** ce cas d'utilisation concerne exclusivement l'administrateur, le statisticien Neema et l'éditeur Neema. Il permet de remplir un formulaire subdivisé en plusieurs onglets incluant l'identification du patient suivi, le résultat du test réalisé ainsi que les traitements et soins. Si le patient est diagnostiqué

positif, les étapes suivantes consistent à renseigner le mode de contamination suspecté, à envoyer une nouvelle fois au laboratoire pour confirmation et vérification du CD4 afin de déterminer le stade clinique de la maladie pour une éventuelle prise en charge. Sinon, la saisie s'arrête à la date du dernier test réalisé. Dans tous les deux cas, les informations sont stockées dans la base.il comprend également la mise à jour et la suppression des données des patients. Sa particularité avec la gestion des patients Sencas est la prise en compte de quelques attributs supplémentaires et également la prise en charge des cas négatifs dans les populations vulnérables.

- ✓ **Gestion des patients Sencas :** il permet la saisie des informations des patients Sencas allant de l'identification aux traitements et soins, à la mise à jour et à la suppression des données des patients. Il ne s'intéresse qu'aux patients diagnostiqués positifs et ne concerne que l'administrateur, le Statisticien Sencas et l'éditeur Sencas.
- ✓ **Consulter liste :** Ce cas permet de voir la liste complète des patients selon son profil. Sauf l'administrateur est autorisé à consulter la liste complète des patients de la base. Les Statisticiens quant à eux, ne sont autorisés à consulter que la liste complète des patients enregistrés par leurs éditeurs respectifs. Le profil éditeur ne consulte que la liste des patients de sa structure.

3.3 Diagramme de classes

Le diagramme de classe est l'un des diagrammes les plus utiles dans la conception car il décrit clairement la structure du système en modélisant ses classes, ses attributs, ses opérations et les relations entre ces objets.

Etant tenu au respect de la structure de la base de données en production, nous voulons signaler au passage que notre modèle n'est pas parti de zéro. Il a été juste généré à partir de l'ancienne base sur laquelle viendra se poser notre applicatif en développement, avec l'outils Database First (vous trouverez un exemple illustratif en annexes). Par ailleurs, pour prendre en compte quelques nouvelles fonctionnalités, nous y avons ajouté quelques tables supplémentaires dont nous montrerons la pertinence plus loin. Il importe de souligner aussi qu'un travail d'assainissement de la base a été fait au préalable avec mon maître de stage avant le démarrage du projet.

Il en ressort quelques captures ci-dessous :

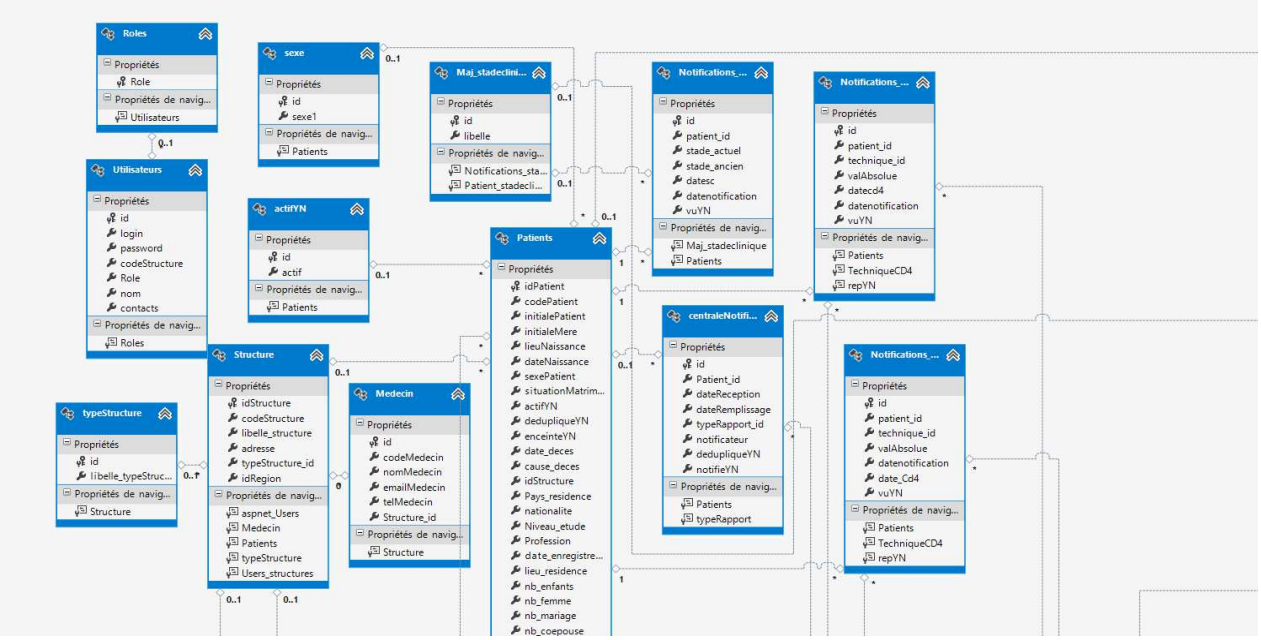


Figure 6:Diagramme de classe

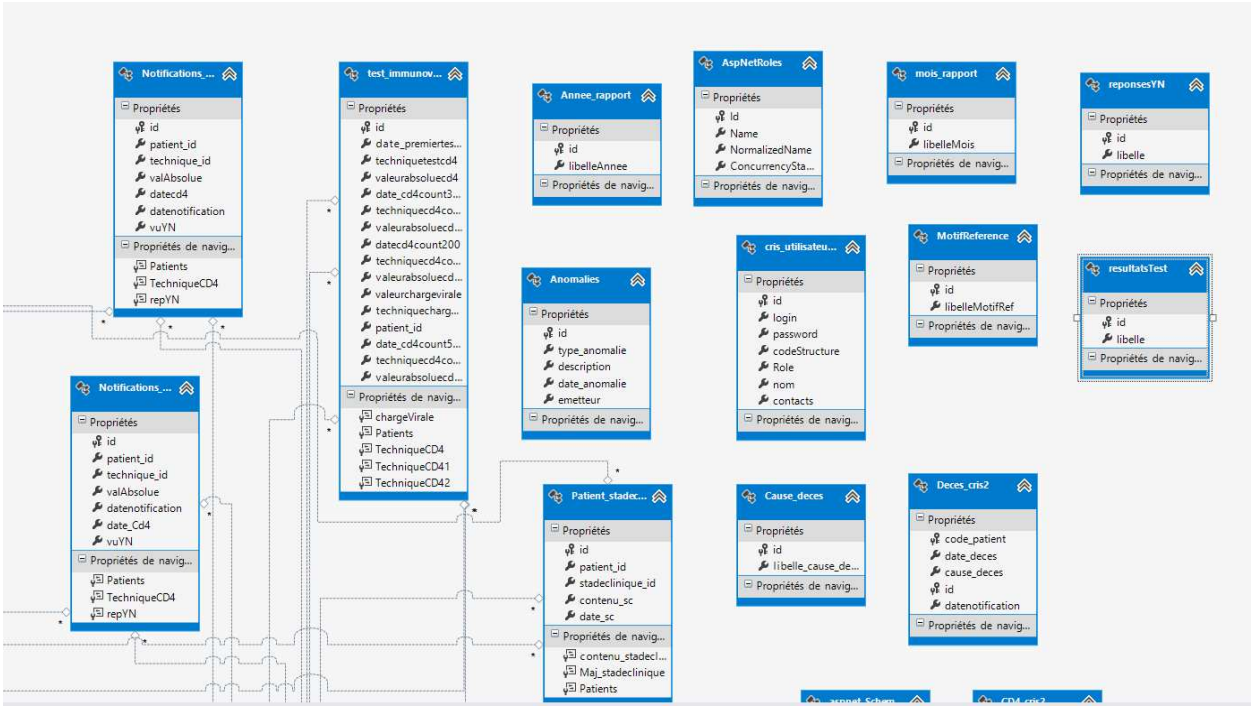


Figure 7:Diagramme de classe

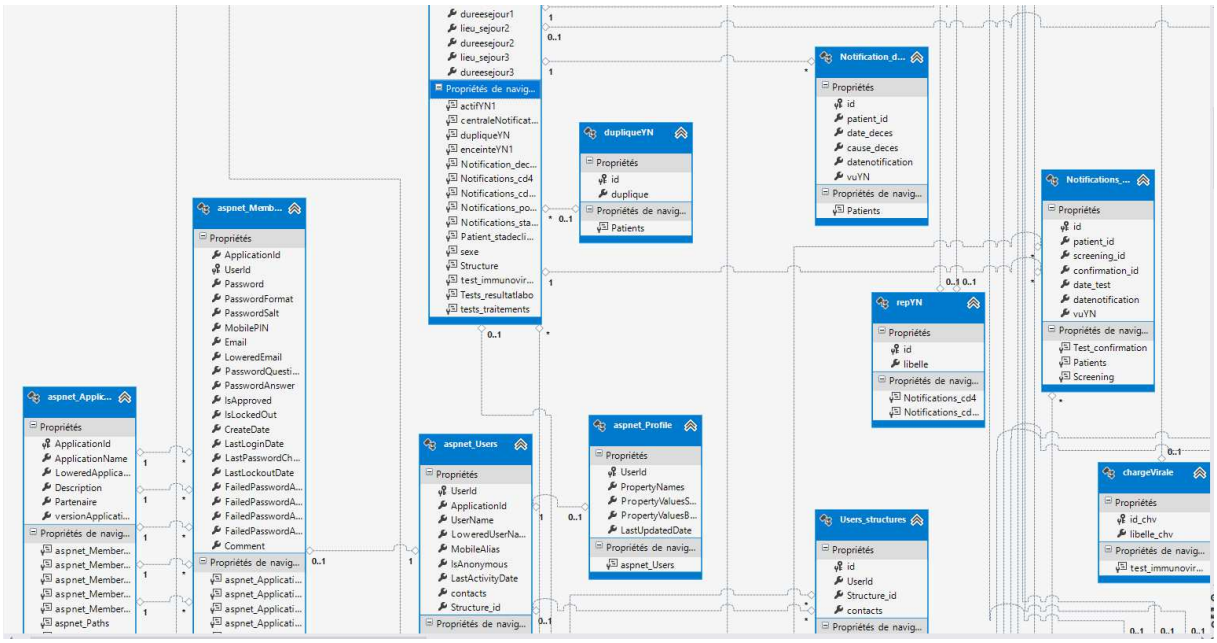


Figure 8:Diagramme de classe

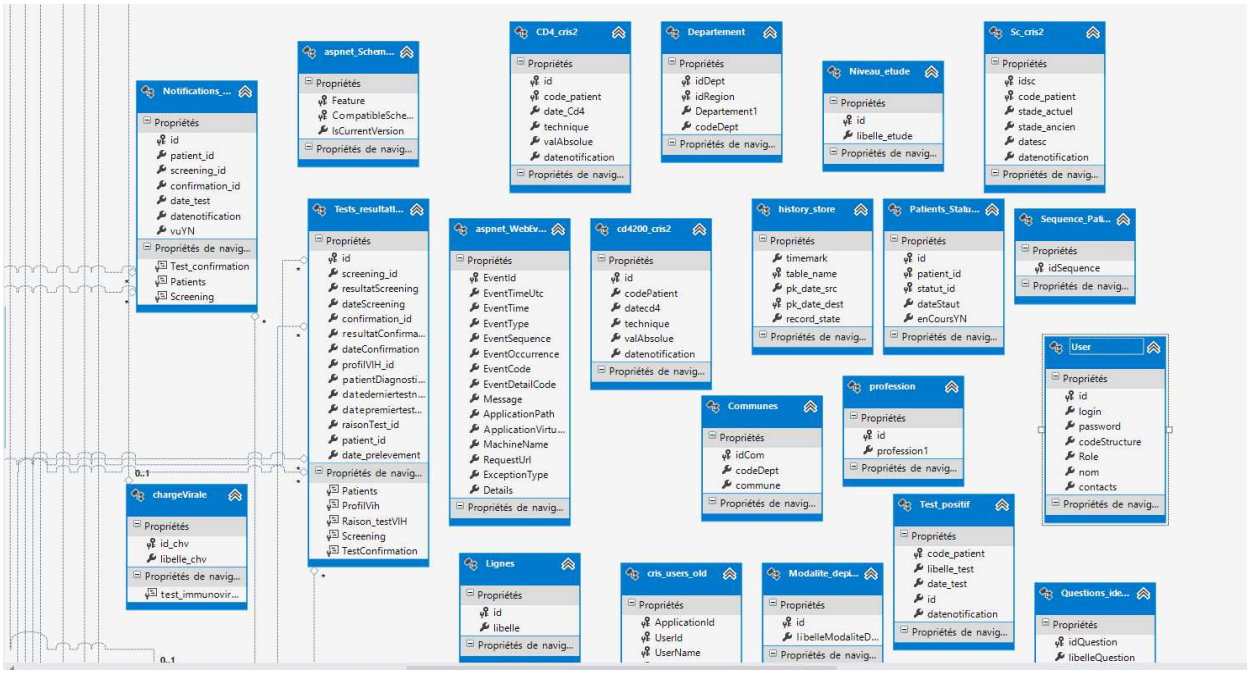


Figure 9:Diagramme de classe

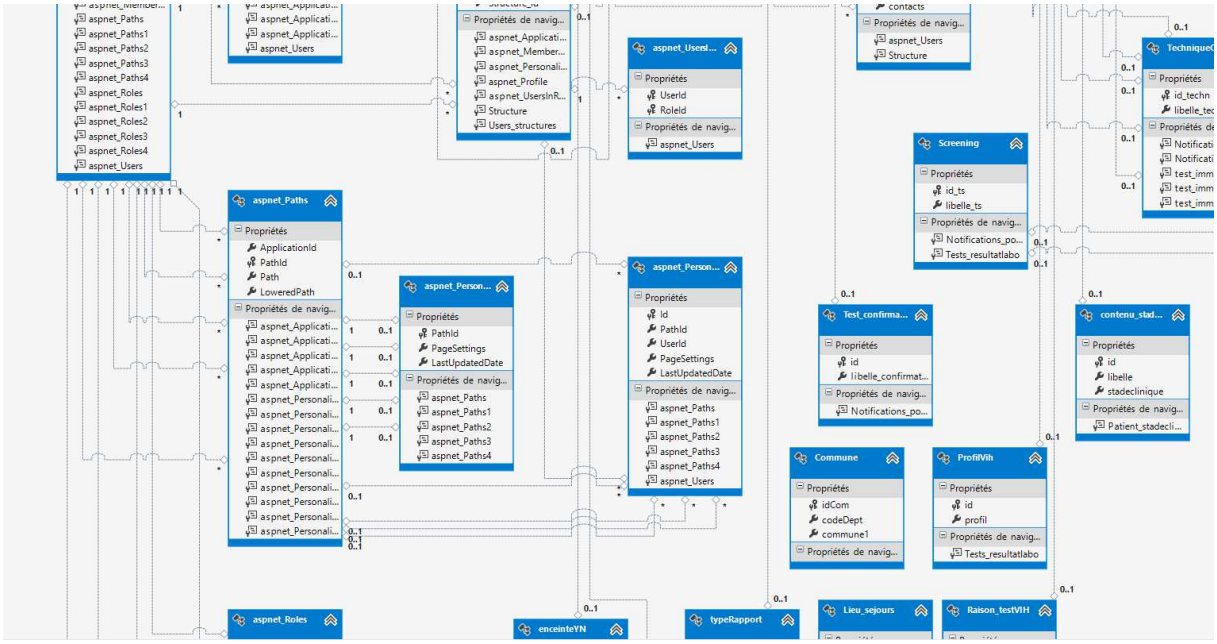


Figure 10:Diagramme de classe



Figure 11:Diagramme de classe

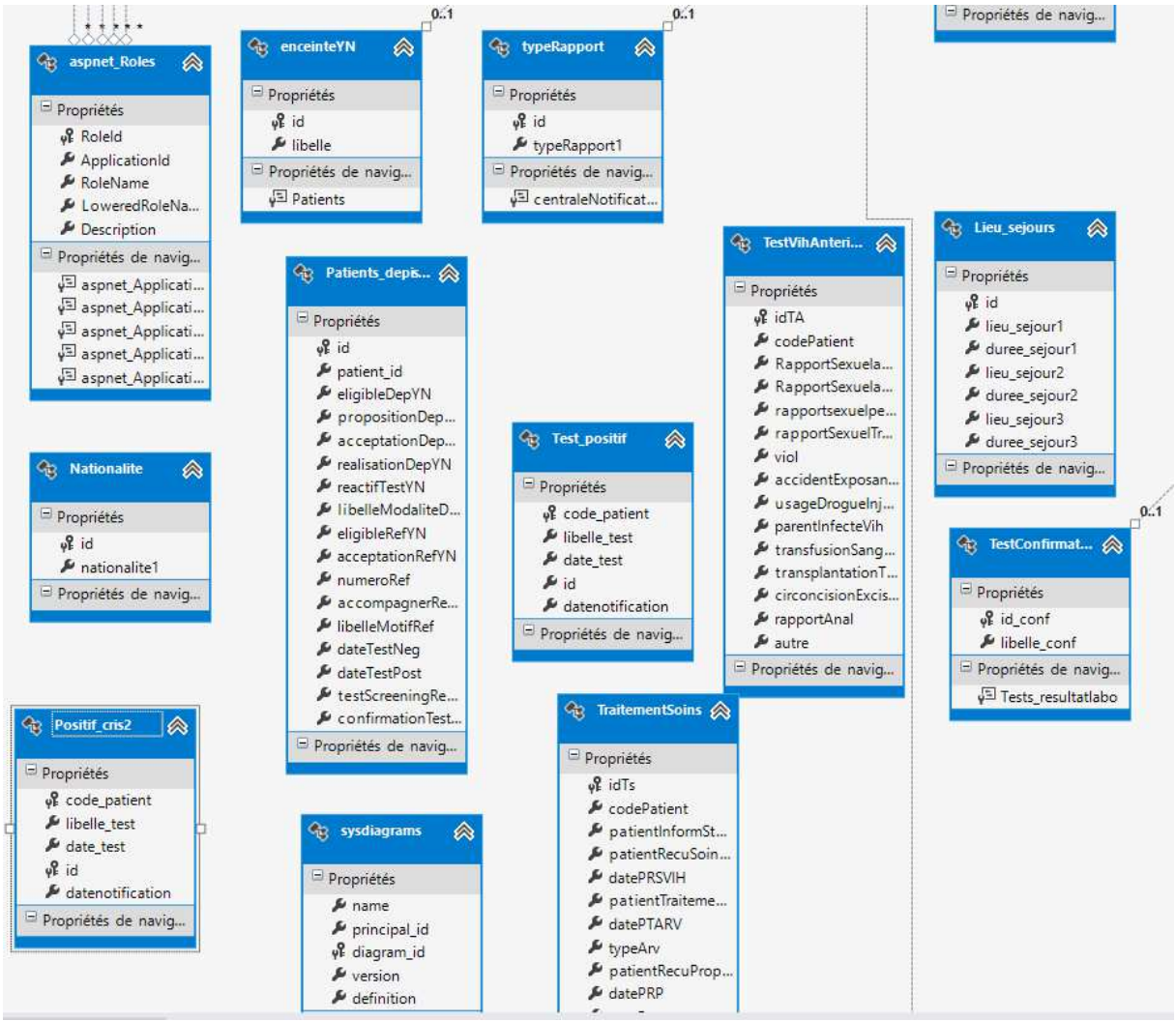


Figure 12:Diagramme de classe

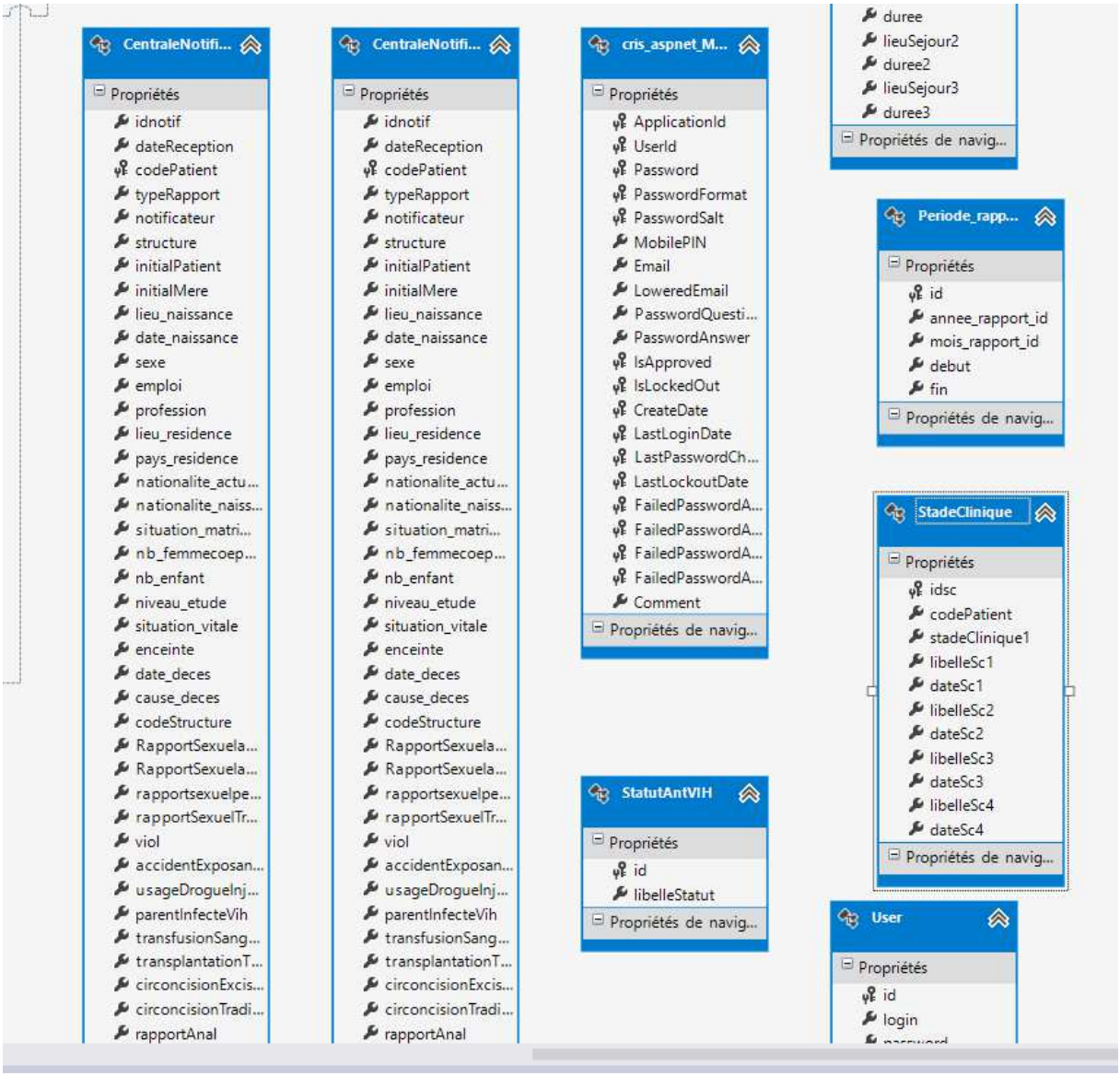


Figure 13:Diagramme de classe

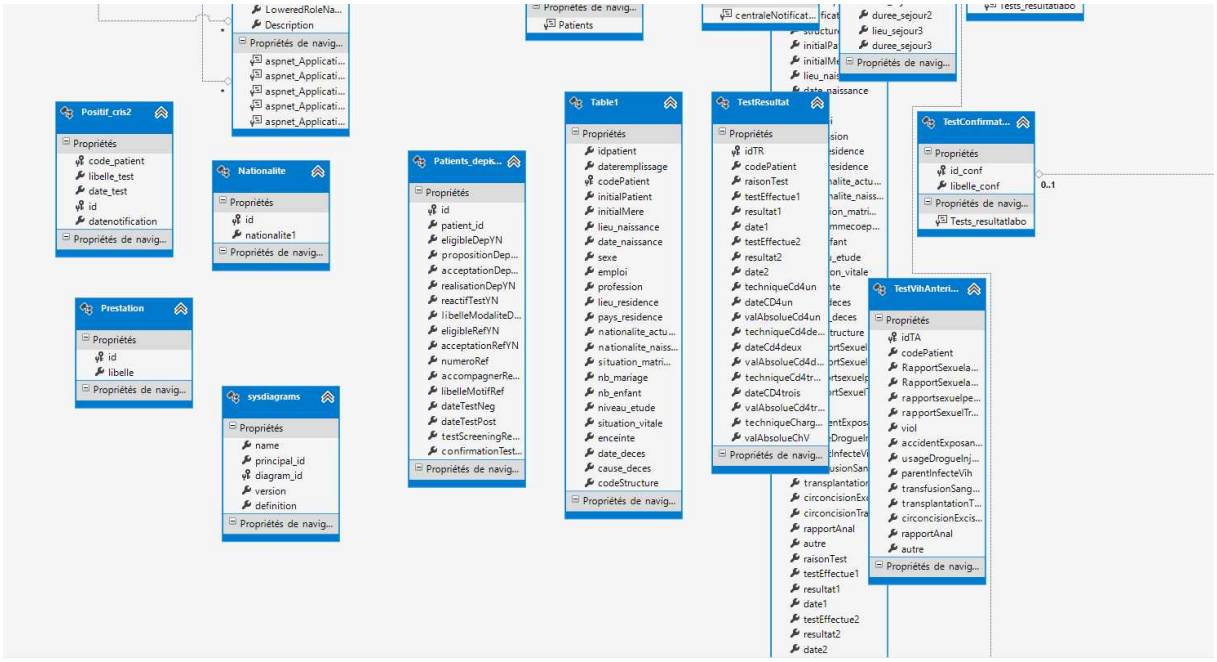


Figure 14:Diagramme de classe

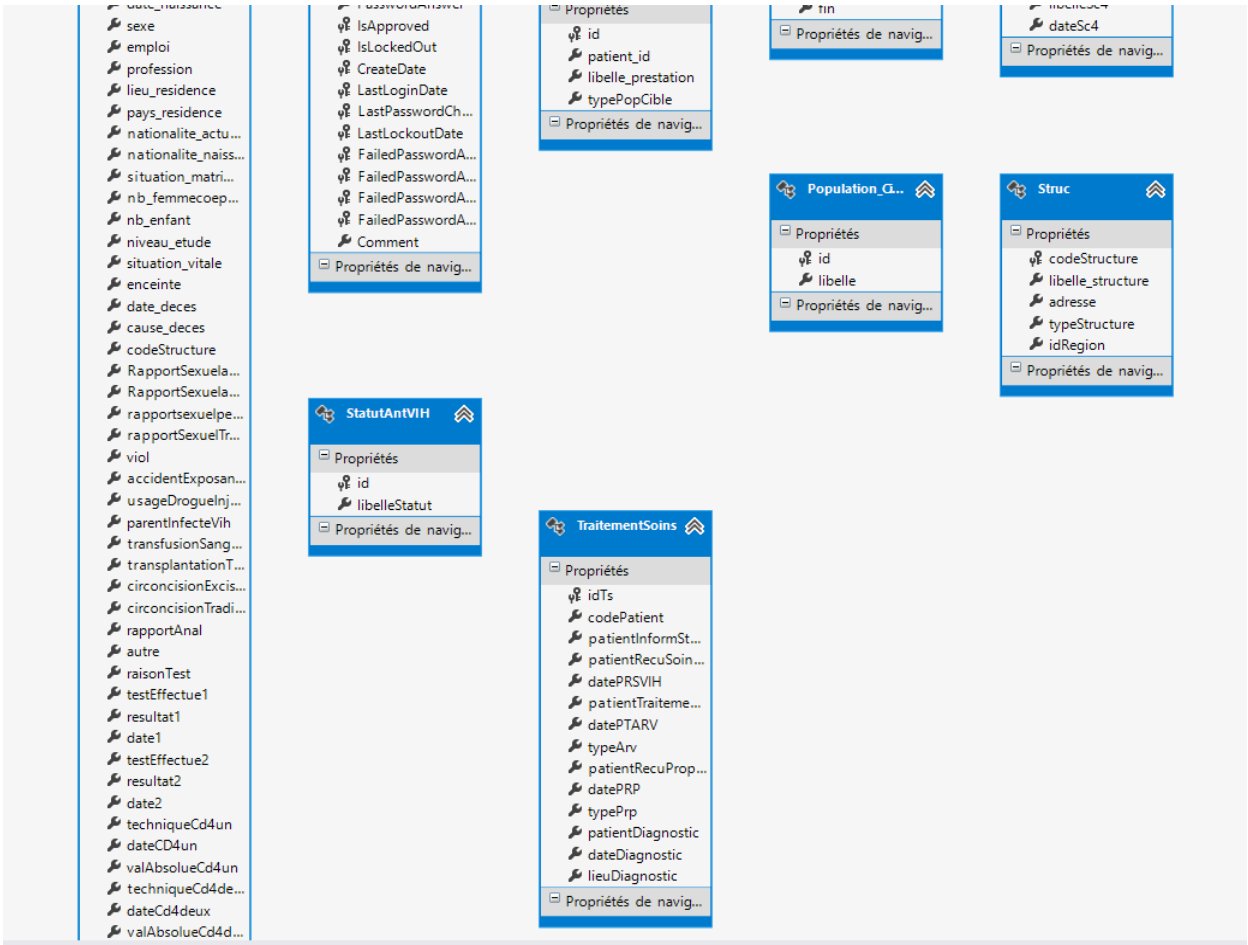


Figure 15:Diagramme de classe

Pour la solution existante, la base de données en production comportait quatre-vingt-neuf tables. Nous en avons nettoyé vingt-cinq et en avons ajouté quelques autres puis avons revu certaines relations pour pouvoir prendre en charge les fonctionnalités de l'Editeur Neema et son Statisticien. Le tableau ci-dessous donne en détail la description des nouvelles tables et leur relation.

Tableau 3 : Description des classes

Table	Propriété	Description	Relation
Patients_depistage	Id	Identifiant de la table	**Patients**
	eligibleDep	Indique l'éligibilité du patient pour le dépistage	
	prpositionDep	Indique si le patient se dépiste suite à une proposition	
	acceptationDep	Indique si oui ou non, le patient a accepté la proposition de dépistage	
	realisationDep	Indique si le dépistage a été effectué ou pas	
	reactifTest	Indique le résultat du test	
	eligibleRef	Indique si le patient est éligible pour avoir un référent	
	acceptationRef	Accepte-t-il un référent	
	libelleModaliteDep	Identifie la modalité du dépistage si test réalisé	
	numeroRef	Numéro de son référent	
Patients_depistage	accompagnerRef	Est-il Accompagné par un référent	**Patients**
	libelleMotifRef	Indique le motif de référence	
	dateTestNeg	Indique la date de la dernière fois ou le test est négatif	
	dateTestPos	Indique la date de la première fois ou le test est positif	
InfoPatientNeema	Id	Identifiant de la table	**Patients**
	libelle_prestation	Indique le lieu de prestation	
	typePopCible	Indique le type de population auquel appartient le patient	

StatutAntVIH	Id	Identifiant de la table	**Patients_StatutAntVIH**
	libelleStatut	Indique la sérologie du patient	
Patients_StatutAntVIH	Id	Identifiant de la table	**Patients Patients_StatutAntVIH**
	dateStatut	Indique la dernière date de réalisation du test	
	encours	Indique la sérologie en cours du patient	
Prestation	Id	Identifiant de la table	Ces deux tables n'ont pas de relation.
	libelle	Indique le nom du lieu de prestation	
Population_Cible	Id	Identifiant de la table	
	libelle	Indique le libellé du type de population	
Patients	Table principale à laquelle est rattachée les autres		

Il ressort de cette description le diagramme de classes ci-dessous :

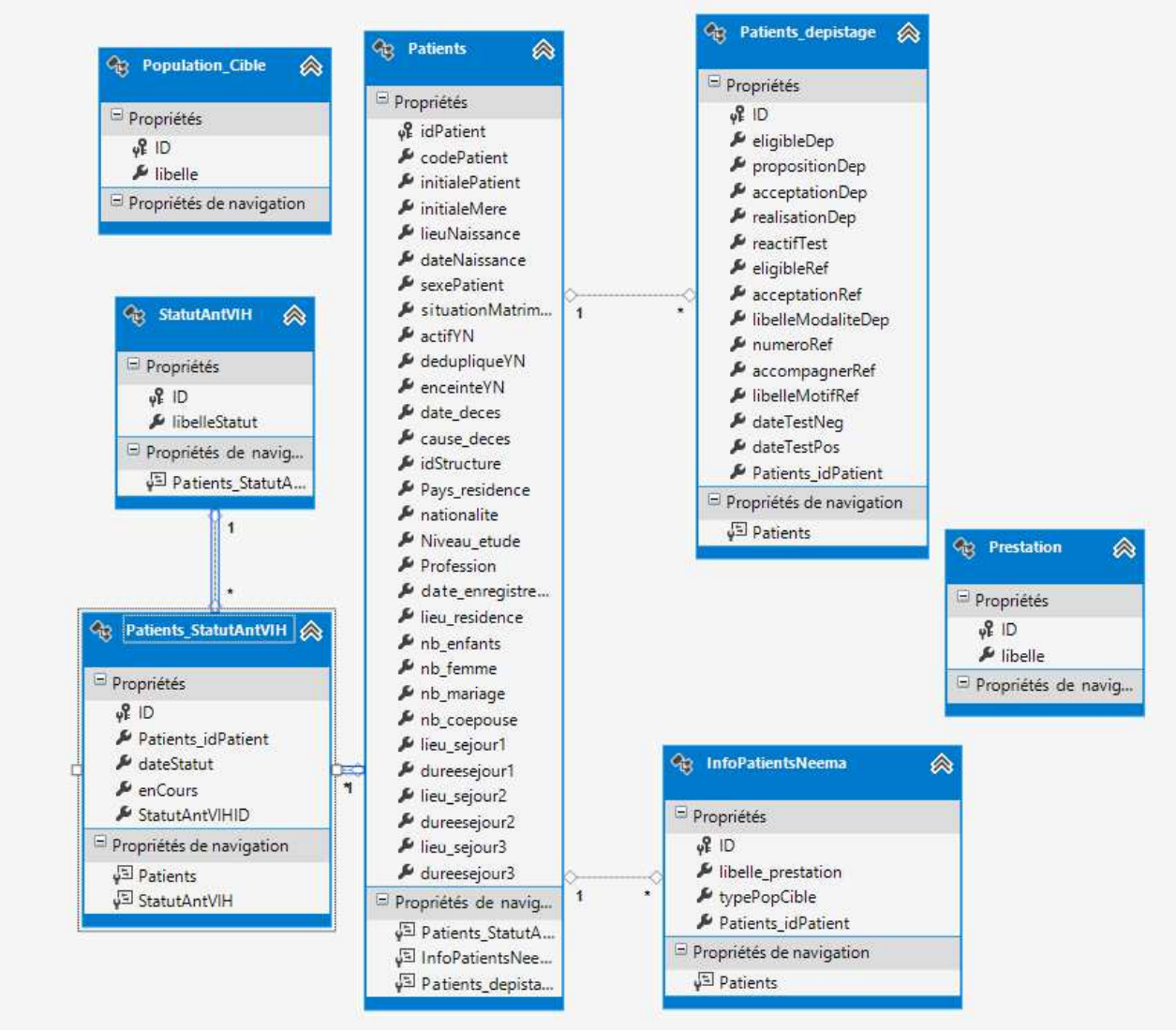


Figure 16: Diagramme de classes ajoutées

Ce diagramme montre l'ajout de cinq nouvelles tables, presque toutes en relation avec la table principale 'Patients'. Certaines nous permettent de récupérer des informations pour nos traitements, d'autres pour le stockage des résultats de ces traitements. Cet ajout permet de prendre en compte les fonctionnalités de l'acteur Neema, qui s'intéresse à une catégorie particulière des données du patient pour ses traitements et prises de décisions éventuelles.

Ce diagramme de classe nous permettra de structurer notre travail de manière efficace.

3.4 Outils et technologies utilisés

3.4.1 Technologies

3.4.1.1 Rappel de ASP.NET Core 2.1

ASP.NET Core 2.1 est un Framework multiplateforme à hautes performances et open source pour créer des applications cloud modernes et connectées à internet.

Sa particularité avec les anciennes versions et la prise en charge de HTTP/2. HTTP2 est une révision majeure de HTTP[6]. Une de ses fonctionnalités remarquables est la prise en charge de la compression d'en-tête et des flux entièrement multiplexés sur une seule connexion.

3.4.1.2 Bootstrap 4.0

Bootstrap est une collection d'outils gratuits et à code source ouvert permettant de créer des sites Web et des applications Web réactifs. Il s'agit du Framework HTML, CSS et JavaScript le plus répandu pour le développement de sites Web réactifs et d'abord mobiles.

Il renferme des outils utiles à la création du design (animation et interactions avec la page, graphisme …).

3.4.1.3 JQuery 3

JQuery est une bibliothèque JavaScript gratuite, libre et multiplateforme, compatible avec l'ensemble des navigateurs web.

Il permet de gagner en rapidité dans les interactions avec le code HTML d'une page Web. La bibliothèque JQuery propose des fonctionnalités suivantes :

- ✓ Manipulation du DOM ;
- ✓ Gestion des événements (clics, mouvements de souris, etc.) ;
- ✓ Effets d'animation ;
- ✓ Etc.

3.4.1.4 CSS 3

CSS est un langage informatique utilisé sur l'internet pour mettre en forme les fichiers HTML ou Xml. Appelés aussi feuilles de style, les fichiers CSS comprennent du code qui permet de gérer le design d'une page en HTML.

3.4.1.5 HTML 5

HyperText Markup Language en abrégé HTML est un langage de base pour la création de site internet. Il sert à créer des documents interopérables avec divers équipements de manière

[6] HTTP : Pour HyperText Transfert Protocol est un protocole de communication client-serveur permettant à un utilisateur d'accéder à des pages web par l'intermédiaire du navigateur.

conforme. HTML est très souvent utilisé avec JavaScript et des feuilles de style en cascade (CSS).

3.4.2 Outils Microsoft Visual Studio

Visual Studio est un ensemble d'outils de développement permettant de générer des applications web ASP.NET, des services web Xml, des applications bureautiques et des applications mobiles

3.4.2.1 Visual Studio 2012 Ultimate

Microsoft Visual Studio 2012 propose une nouvelle génération d'outils de développement, de collaboration et de gestion de cycle pour la conception des applications.

Il a été créé depuis le septembre 2012 et se décline en plusieurs éditions (Ultimate, Professionnel, Premium, etc.), destinées à des profils différents. Grâce à ses outils de conception, nous avons mis en place nos différents diagrammes.

3.4.2.2 Visual Studio 2019 Community

Microsoft Visual Studio 2019 Community est un environnement de développement intégré (IDE) extensible, complet et gratuit pour créer des applications modernes pour Windows, Android et iOS, ainsi que des applications web et des services cloud. Il renferme des outils de codage, débogage, web, des applications de périphérie, l'intégration de Git et d'autres avantages fabuleux.

Ces outils permettent :

- ✓ De coder, déceler et résoudre rapidement les erreurs en toute simplicité ;
- ✓ Déboguer pour corriger rapidement les bogues ;
- ✓ Créer des applications web modernes ;
- ✓ Héberger du code partout ;
- ✓ Etc.

3.5 Implémentation de SENCAS

Dans cette partie, nous présenterons les interfaces de quelques fonctionnalités opérationnelles de SENCAS version 2.0. Il est à noter que pour y accéder, il faut au préalable s'authentifier avec un compte utilisateur créer par l'administrateur du système. Après authentification, l'utilisateur aura une redirection vers son tableau de bord, selon son profil.

La figure ci-dessous présente la page d'authentification. Elle permet à un utilisateur déjà inscrit de s'authentifier en utilisant son login et son mot de passe fournis à l'inscription.



Figure 17:page d'authentification de la plateforme

La figure 18 montre le tableau de bord de l'administrateur. Nous y observons toutes les options de la plateforme.

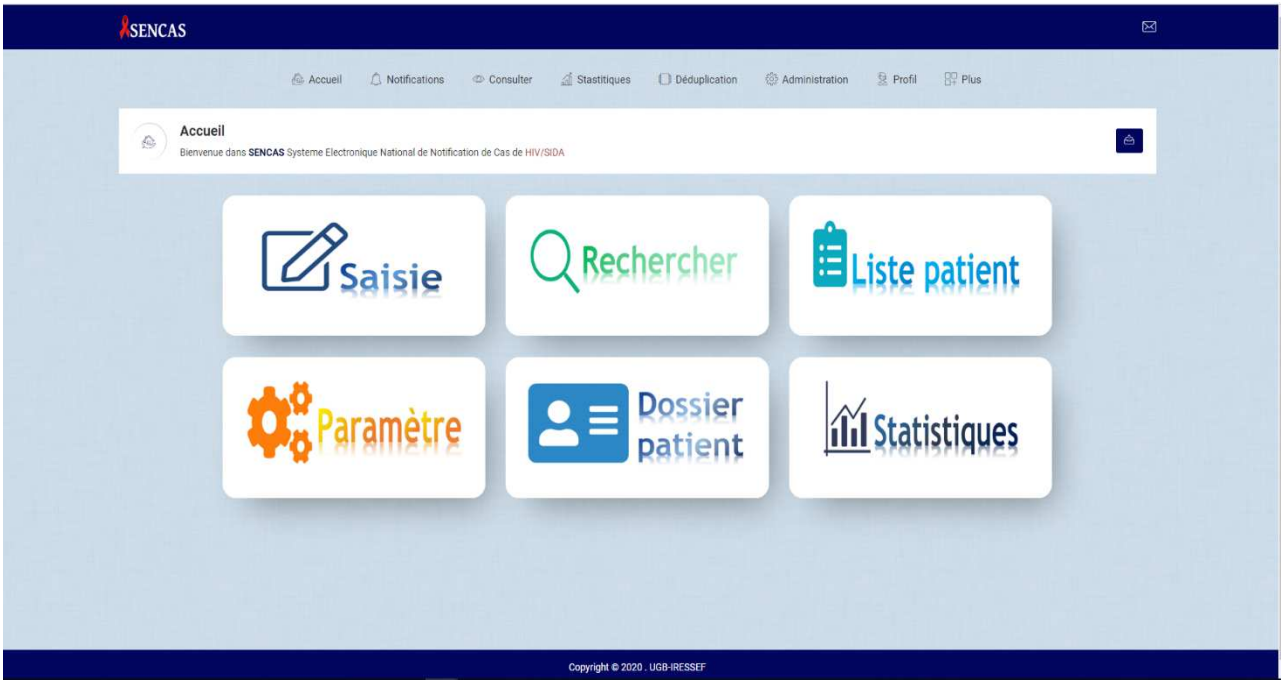


Figure 18:Tableau de bord de l'administrateur du système

3.5.1 Saisie

Ce module affiche un formulaire à onglets pour l'identification du patient et d'autres informations spécifiques pour sa prise en charge.

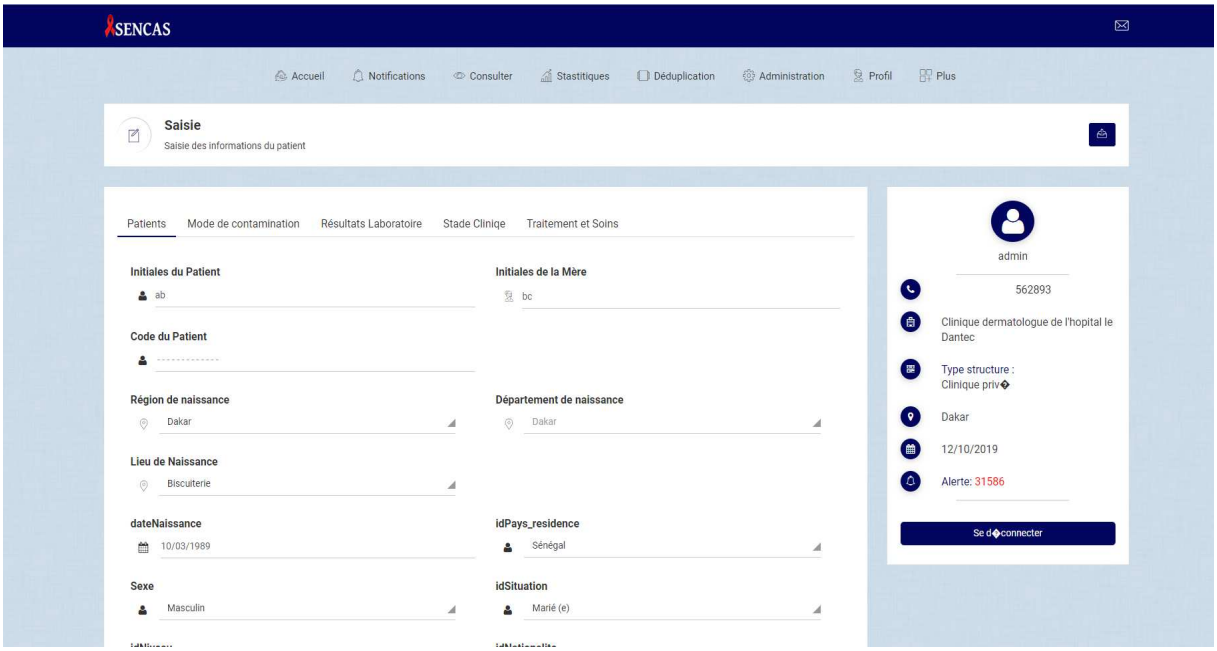


Figure 19:Formulaire de saisie

3.5.2 Liste

La liste des patients contient quelques informations d'un patient par ligne. Pour visualiser les informations complètes, il suffit de cliquer sur le bouton blanc.

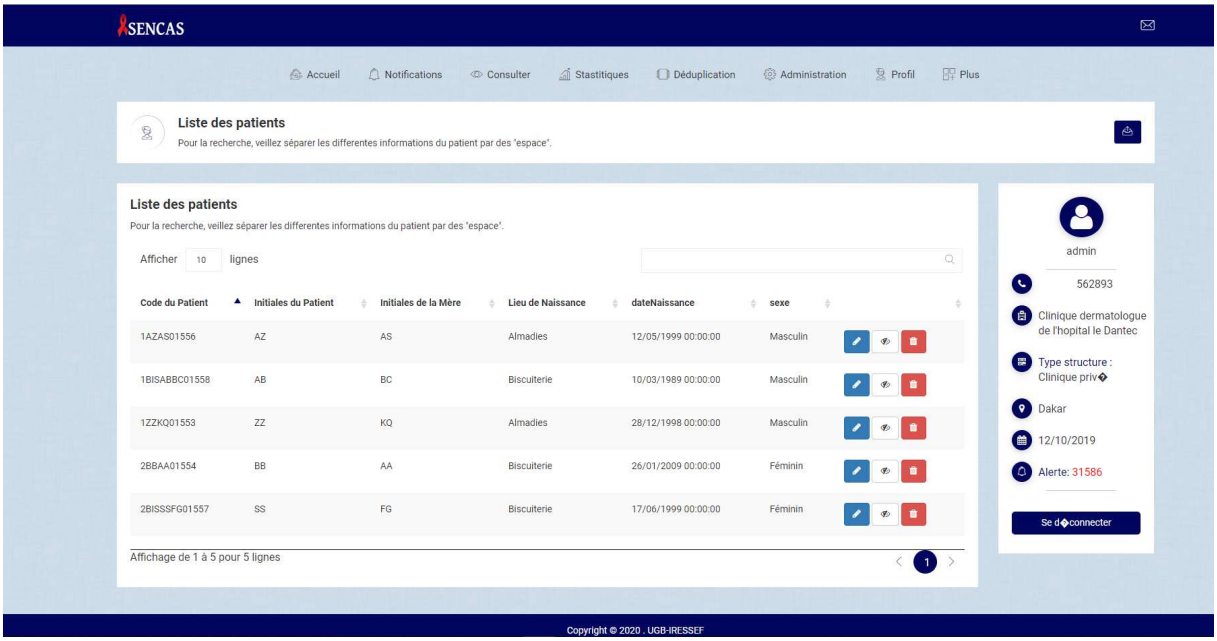


Figure 20:Liste des patients

Conclusion

Dans le cadre de notre stage, la mission qui nous a été confié est en cours et nous comptons la poursuivre jusqu'à son accomplissement.

Pour le moment, tout le travail de conception et d'implémentation de certaines fonctionnalités ont été réalisées. Nous pencherons nos travaux futurs sur les fonctionnalités de l'acteur Neema et son statisticien.

Bien que le projet soit en cours de développement, il nous a permis d'acquérir de nombreuses connaissances, notamment dans le domaine du développement d'application web avec le Framework .Net Core via ASP.NET MVC. Nous avons pu également découvrir la vie au quotidien en entreprise.

Sur le plan personnel, notre niveau de programmation en C# et notre connaissance du Framework .NET a considérablement augmenté par rapport aux acquis universitaires.

Ce stage nous a aussi permis d'assimiler des connaissances dans le déroulement du processus de développement en équipe et quelques autres réalités pratiques d'entreprise. Ce fut une expérience très enrichissante qui nous introduit dans le monde du travail.

A l'avenir, nous souhaiterons finaliser le projet en vue de nous perfectionner d'avantages dans le développement d'application Web.

Bibliographie et webographie

Références bibliographiques

A. Archives, CCOS
B. Jeffrey Richter, Programmer Microsoft .NET Framework

Références webographiques

[1]. https://fr.wikipedia.org/wiki/Framework_.Net

[2]. https://www.softfluent.fr/blog/architecture-logicielle-pour-application/

[3]. https://dotnet.microsoft.com/apps/aspnet

[4]. https://www.dotnetdojo.com/8-differences-a-connaitre-entre-asp-net-webforms-et-asp-net-mvc/

[5]. https://www.dotnettricks.com/learn/mvc/difference-between-aspnet-webform-and-aspnet-mvc

[6]. https://www.synbioz.com/blog/architecture-applications-web

[7]. https://www.developpez.net/forums/d1177445/dotnet/developpement-web-net/asp-net/webforms-vs-mvc-lequel/

[8]. https://openclassrooms.com/fr/courses/1730206-apprenez-asp-net-mvc

[9]. https://www.w3schools.com/asp/

Annexes

Génération du diagramme de classe avec ADO.NET

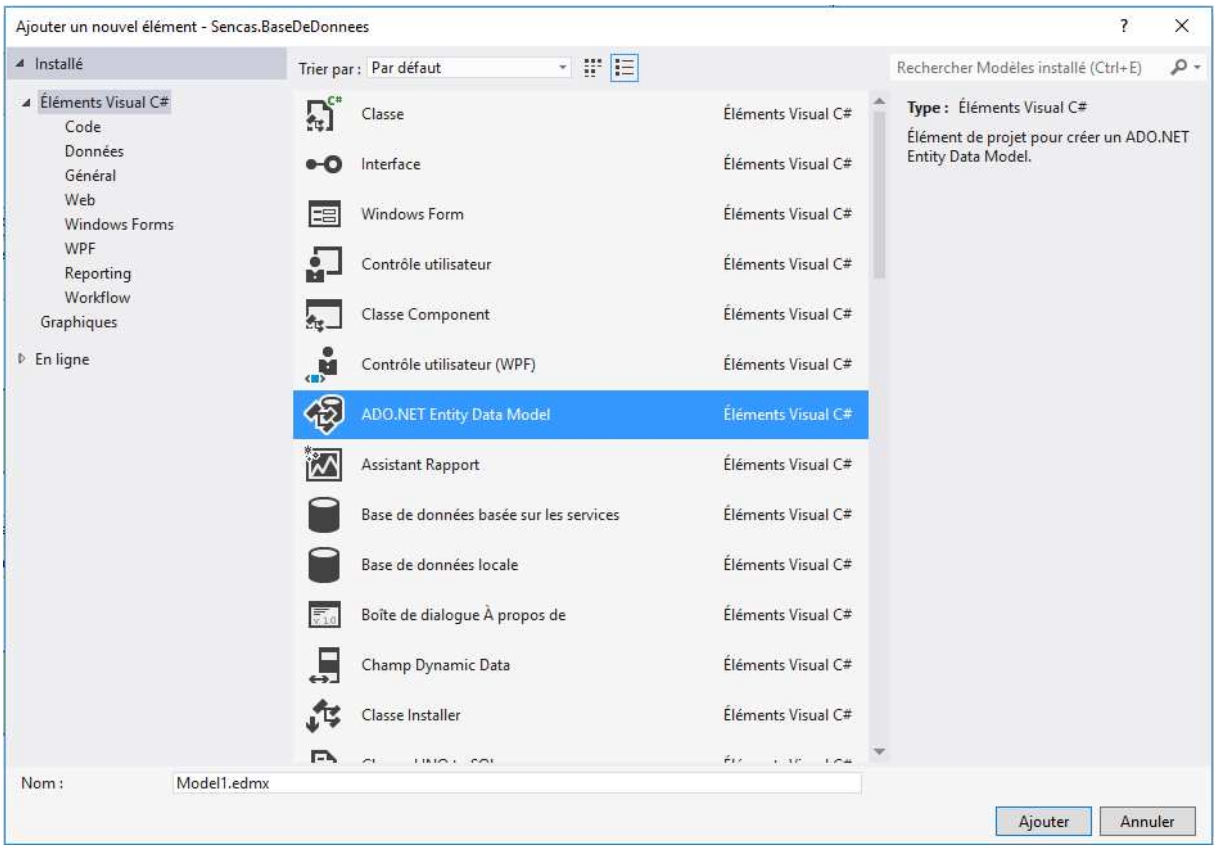


Figure 21:Outils de génération

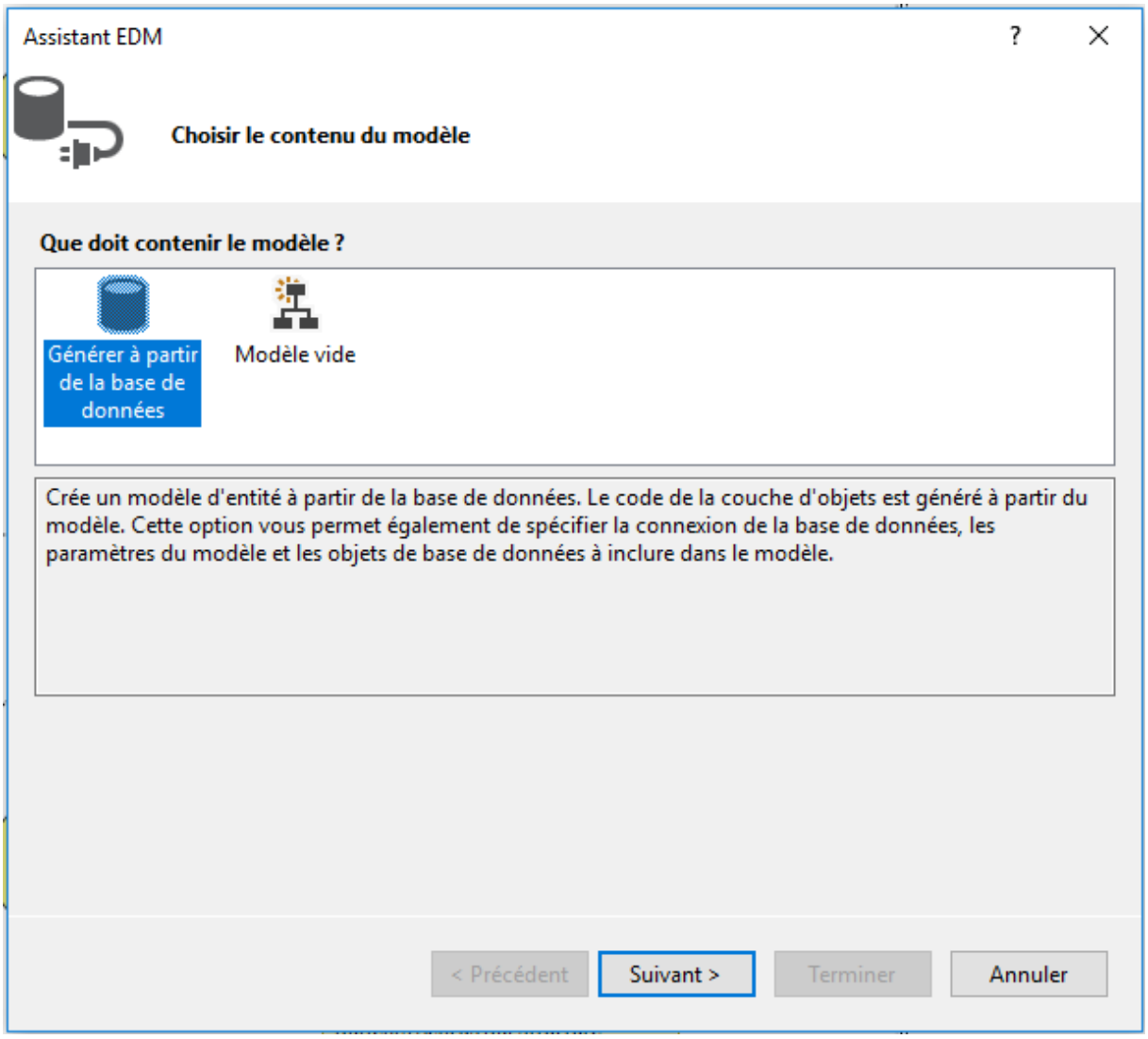


Figure 22:Contenu du Modèle

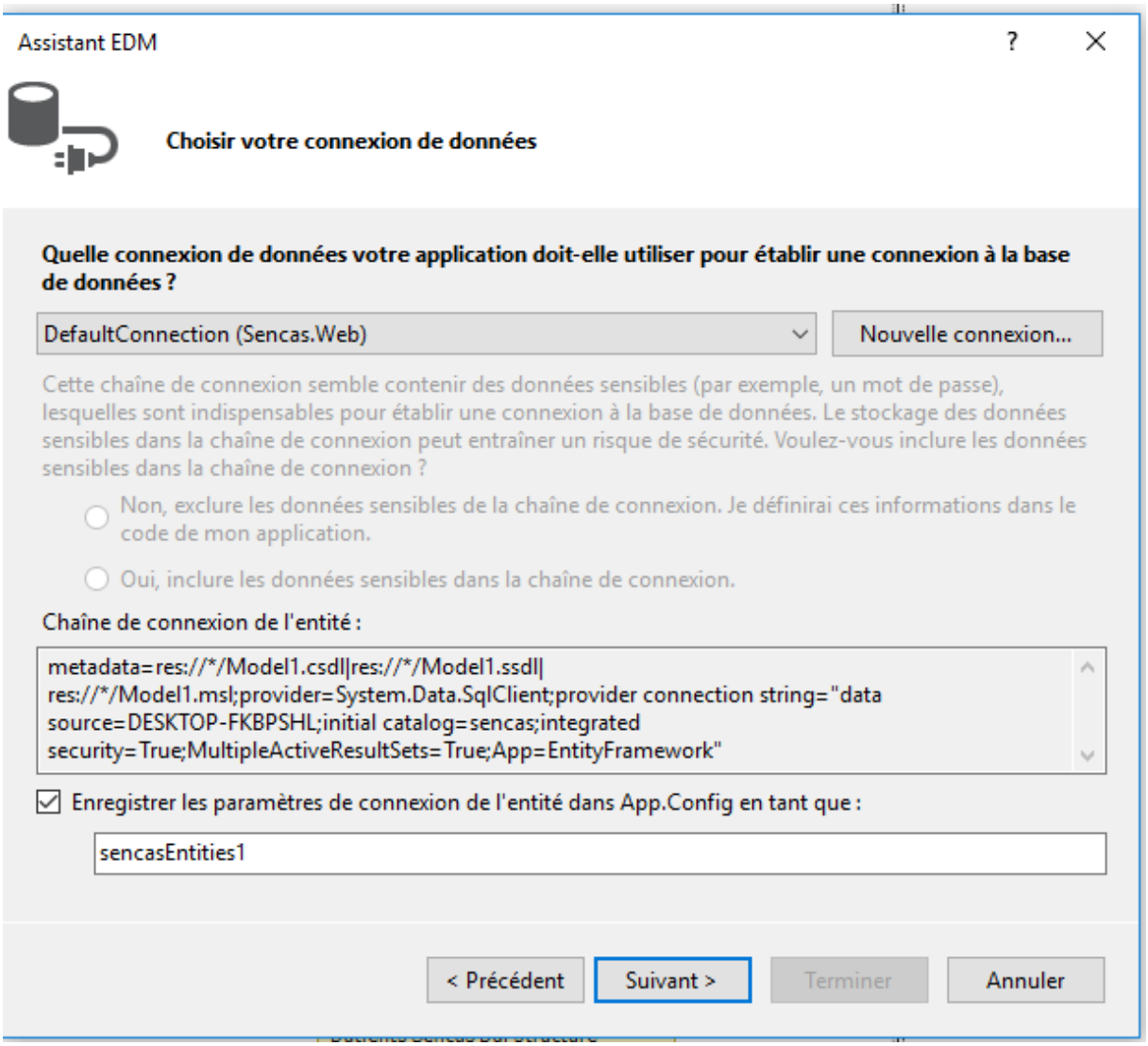


Figure 23:Choix de la connexion

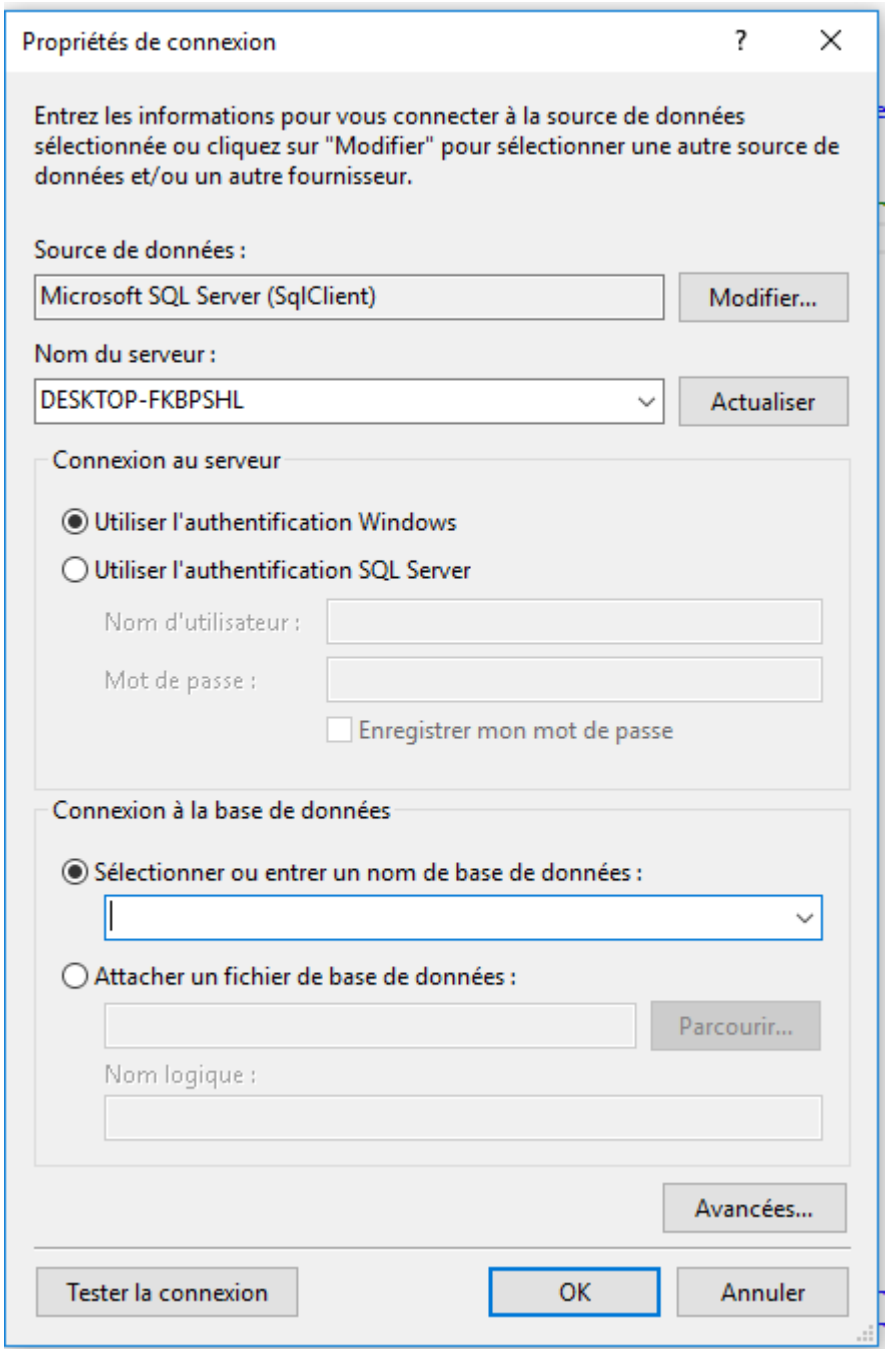


Figure 24:Choix de la base

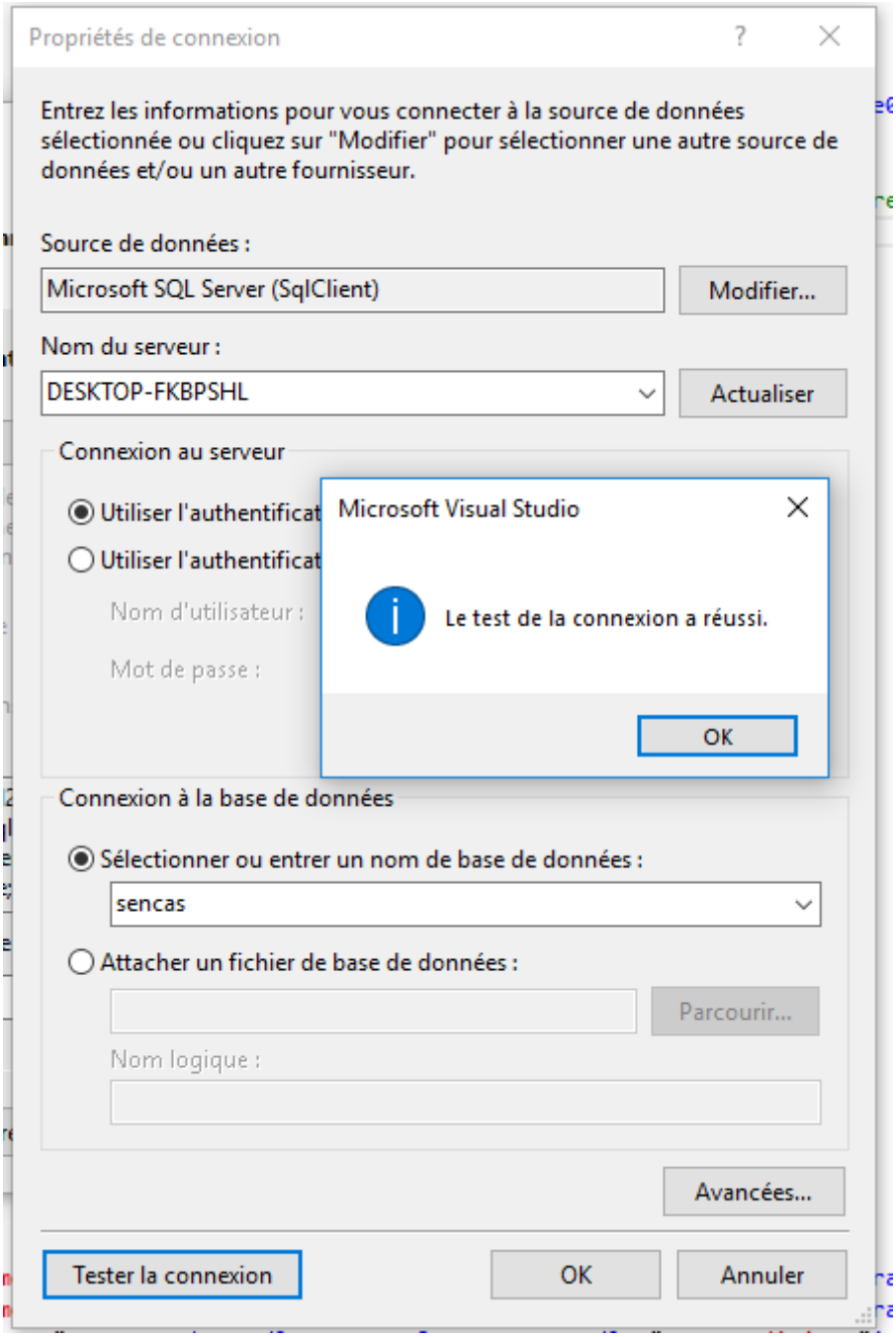


Figure 25:Test de connexion

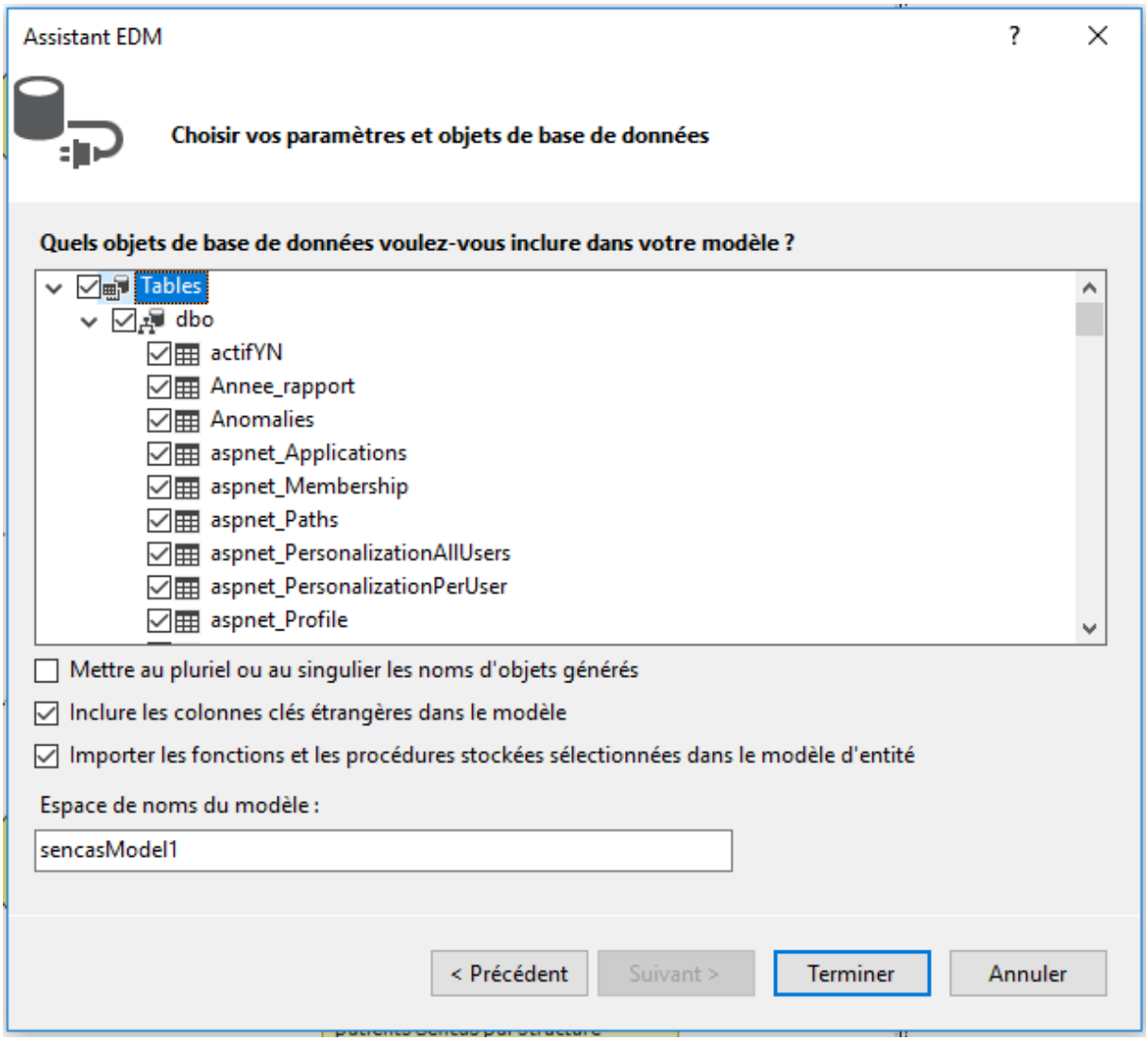


Figure 26:Choix des tables, des vues et des procédures stockées