Martines Stefano
Magnin Adrien

AbsMedical - MARTINES / MAGNIN

Sommaire

Martines Stefano
Magnin Adrien

1. Le projet

1.1. Objectif

AbsMedical est une application bureau à destination des médecins dont le but est simple:

Justifier numériquement les absences scolaires d'un étudiant.

1.2. Principe

Le principe fonctionne comme suit:
L'étudiant est malade et décide donc d'aller chez son médecin traitant (ou non) pour se faire ausculter. Au lieu de demander un certificat médical manuscrit, l'étudiant peut s'il le souhaite, demander un certificat numérique. S'il n'a pas de carte étudiante, le médecin doit établir alors un certificat médical manuscrit.
Le médecin ouvre alors son logiciel "AbsMedical" et vois les informations concernant l'étudiant (nom, prénom, numéro sécurité social, adresse postal, école correspondante, ...) et clique sur le bouton permettant de renseigner une absence pour l'étudiant en cours, en y indiquant la date de visite, date de début d'absence, date de fin d'absence (si absence scolaire sur plusieurs jours) et le motif.
Une fois que le médecin a enregistré l'absence, le logiciel enregistre en base de données l'absence et notifie par email l'école de l'étudiant ainsi que l'étudiant si précisé.
De plus un PDF peut être généré et enregistré sur l'ordinateur du médecin où cas ou il aurait besoin d'une copie.

1.3. Fonctionnalités

Le logiciel repose sur quatre fonctionnalités majeures:

- L'enregistrement d'un certificat médical,
- La gestion des étudiants (ajout, modification et suppression),
- La gestion de son profil,
- La consultation de l'historique des absences étudiante.

Martines Stefano
Magnin Adrien

1.4. Avantages et inconvénient

- Evite la paperasse tant pour l'école que le médecin.
- L'école notifiée par mail en temps réel.
- Certification de l'absence via l'authentification du médecin.

- Impossible de rédiger une absence par le biais de ce logiciel sans la carte étudiant.

2. Equipe et répartition des tâches

2.1. L'équipe

Le projet a été développé par:

- MARTINES STEFANO
- MAGNIN ADRIEN

2.2. Répartition des tâches

	MARTINES STEFANO	MAGNIN ADRIEN
Conception de la base de données	X	X
Architecture logicielle	X	
Mise en place du model Entity Framework	X	
Fenêtre - Certificat medical		X
Fenêtre - Connexion		X
Fenêtre - Home	X	
Fenêtre - Historique		X
Fenêtre - Ajout étudiant	X	
Fenêtre - Edition étudiant	X	
Fenêtre - Gestion des étudiants	X	
Encryptions des mots de passes		X
Ecriture des requêtes communiquant avec la BDD	X	X
Mise en place des différents services web WCF	X	X
Ecriture de la classe pour la création et l'enregistrement des PDF	X	
Ecriture de la classe pour l'envoi de mail		X

Martines Stefano
Magnin Adrien

3. L'application

3.1. Type d'application

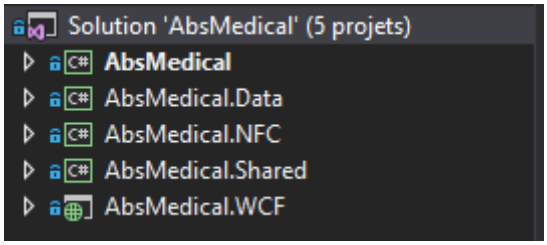
Il s'agit d'une application bureau de type client lourd développé en WPF (**Windows Presentation Foundation)**.

3.2. Technologies utilisées

Langage de programmation: C#
Framework .NET: 4.6.1
IDE: Visual Studio 2015 Community
Base de données: MySQL
Entity Framework: 6.0.0
UI: Windows & MahApps (http://mahapps.com/)
Service web WCF de type SOAP

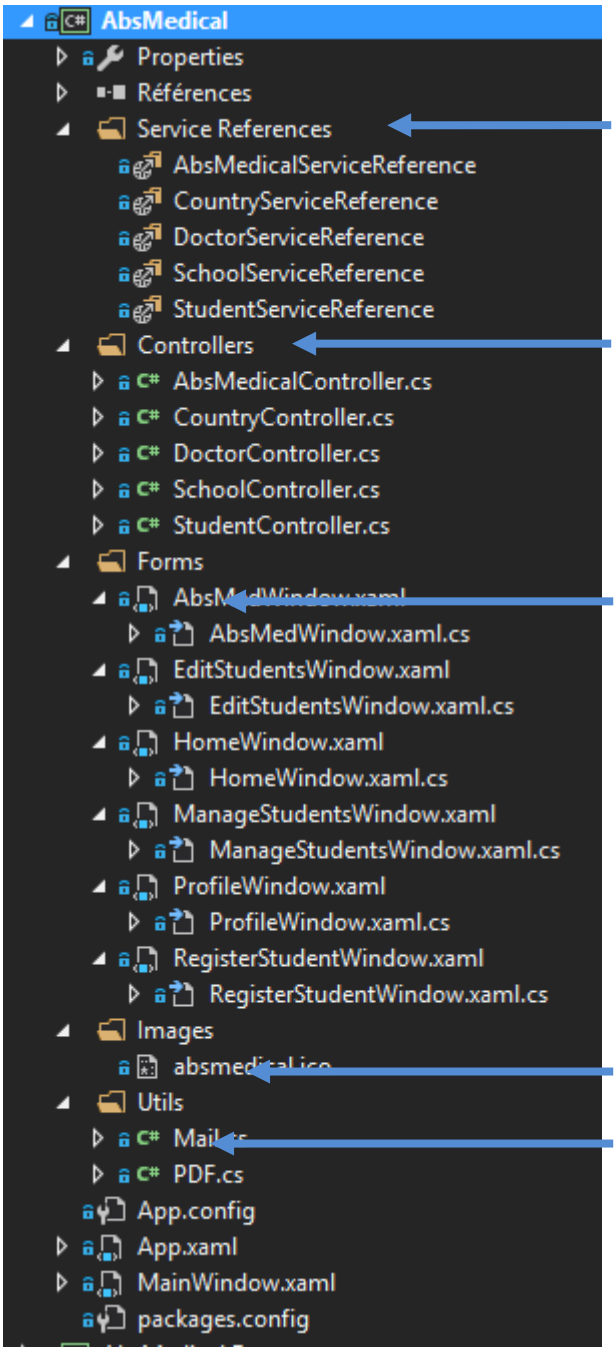
3.3. Architecture logicielle

L'application AbsMedical est répartit en 5 projets.



3.3.1. AbsMedical

Application WPF. Contient l'application client lourd.



Service References: Contient les différentes références de service créé dans le projet AbsMedical.WCF

Controllers: Contient les classes interrogeant les différents services web du projet AbsMedical.WCF.

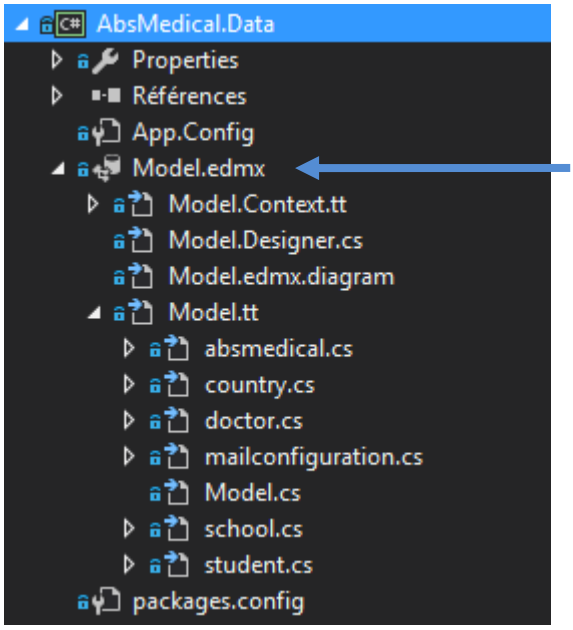
Forms: Contient les designs (.xaml) des différents Fenêtres ainsi que leur code respectif (.xaml.cs).

Images: Contient les différentes images utilisées par l'application.

Utils: Contient les différentes classes utiles a l'application.

3.3.2. AbsMedical.Data

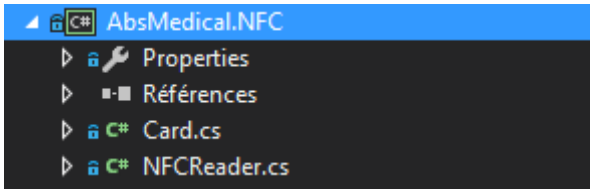
Bibliothèque de Classes. Contient le model de données et les entités.



Model.edmx: Contient la représentation graphique ainsi que les différentes entités de la base de données.

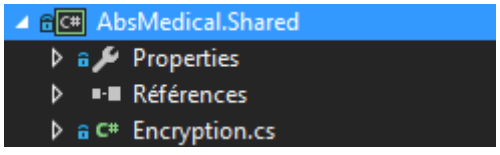
3.3.3. AbsMedical.NFC

Bibliothèque de Classes. Contient le système de lecture de puce NFC.



3.3.4. AbsMedical.Shared

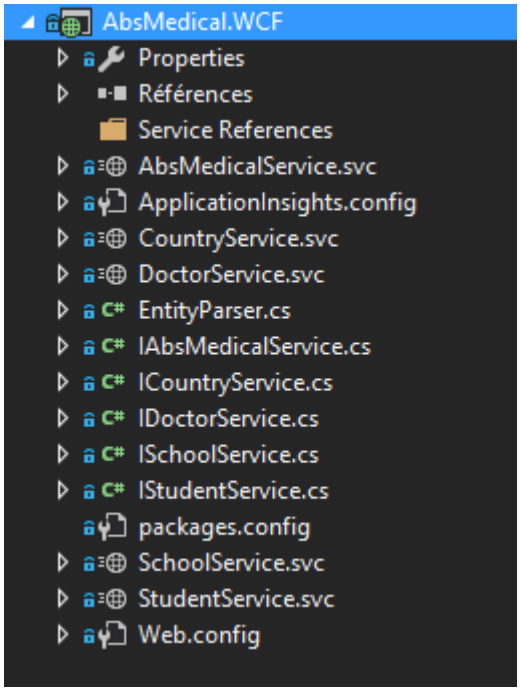
Bibliothèque de Classes. Contient les classes partagées dans toute la solution.



3.3.5. AbsMedical.WCF

Bibliothèque de Services WCF. Contient les différents web services WCF.

Les services web WCF sont:
- AbsMedicalService
- CountryService
- DoctorService
- SchoolService
- StudentService



3.4. Configuration de l'application

Le seul élément configurable dans l'application pour que celle-ci soit opérationnelle est la chaine de connexion présente dans ces différents fichiers:

- AbsMedical/App.config
- AbsMedical.Data/App.config
- AbsMedical.WCF/Web.config

```
  <connectionStrings>
    <add name="rfidEntities"
connectionString="metadata=res://*/Model.csdl|res://*/Model.ssdl|res://*/Model.msl;pro
vider=MySql.Data.MySqlClient;provider connection string=&quot;server=localhost;user
id=root;database=rfid;persistsecurityinfo=True&quot;"
providerName="System.Data.EntityClient" />
  </connectionStrings>
```

3.5. Les services web

Les services web créés pour l'application sont de type SOAP (Simple Object Access Protocol) et utilisent l'infrastructure WCF (Windows Communication Foundation).
Les services web sont accessibles dans le projet AbsMedical.WCF.

Chaque service dispose de son interface et d'une classe implémentant celle-ci.
De plus, pour chaque entité de la base de données, une nouvelle classe (plus light) a été développé afin de ne pas retourner un objet avec des propriétés inutiles.

Pour les requêtes en base de données, le framework Entity Framework ainsi que LinQ ont été utilisés permettant ainsi une meilleure lisibilité dans le code et une meilleure manipulation des données.

3.6. Entity Framework

Entity Framework est un outil permettant de créer une couche d'accès aux données lié à une base de données.

Il propose ainsi la création d'un schéma conceptuel composé d'entités qui permettent la manipulation d'une source de données, **sans écrire une seule ligne de SQL**, grâce à LinQ To Entities.

Martines Stefano
Magnin Adrien

3.6.1. Exemple de service web utilisant Entity Framework

IStudentService.cs

```
 [ServiceContract]
public interface IStudentService
{
    [OperationContract]
    Student GetStudent(string StudentGuid);

    [OperationContract]
    bool RegisterStudent(Student student);

    [OperationContract]
    Student GetStudentBySocialSecurityNumber(string value);

    [OperationContract]
    Student GetStudentByFilters(string firstname, string lastname, DateTime birthdate);

    [OperationContract]
    bool UpdateStudent(Student student);

    [OperationContract]
    bool DeleteStudent(string studentGuid);

}

[Serializable]
[DataContract]
public class Student
{
    [DataMember]
    public string Guid { get; set; }
    [DataMember]
    public string Firstname { get; set; }
    [DataMember]
    public string Lastname { get; set; }
    [DataMember]
    public string Email { get; set; }
    [DataMember]
    public string Phone { get; set; }
    [DataMember]
    public DateTime Birthdate { get; set; }
    [DataMember]
    public string Birthplace { get; set; }
    [DataMember]
    public string StudentId { get; set; }
    [DataMember]
    public string SocialSecurityNumber { get; set; }
    [DataMember]
    public string Address { get; set; }
    [DataMember]
    public string PostalCode { get; set; }
    [DataMember]
    public string City { get; set; }
    [DataMember]
    public string CountryId { get; set; }
    [DataMember]
    public string SchoolGuid { get; set; }
    [DataMember]
    public string DisplayedName { get; set; }
}
```

Martines Stefano
Magnin Adrien

StudentService.svc.cs

```
public class StudentService : IStudentService
    {
        public Student GetStudent(string StudentGuid)
        {
            using (rfidEntities db = new rfidEntities())
            {
                var studentEntity = db.student.FirstOrDefault(i => i.Guid == StudentGuid);
                if (studentEntity != null) return EntityParser.EntityToObject(studentEntity);
                else return null;
            }
        }

        public Student GetStudentBySocialSecurityNumber(string value)
        {
            using (rfidEntities db = new rfidEntities())
            {
                var studentEntity = db.student.FirstOrDefault(s => s.SocialSecurityNumber == value);
                if (studentEntity != null) return EntityParser.EntityToObject(studentEntity);
                else return null;
            }
        }

        public bool DeleteStudent(string studentGuid)
        {
            using (rfidEntities db = new rfidEntities())
            {
                var studentEntity = db.student.FirstOrDefault(s => s.Guid == studentGuid);
                db.student.Remove(studentEntity);
                int result = db.SaveChanges();
                return result > 0;
            }
        }

        public Student GetStudentByFilters(string firstname, string lastname, DateTime birthdate)
        {
            List<Student> students = new List<Student>();
            using (rfidEntities db = new rfidEntities())
            {
                var studentEntity = (from s in db.student where s.Firstname.Contains(firstname) &&
s.Lastname.Contains(lastname) && s.Birthdate == birthdate select s).FirstOrDefault();

                if (studentEntity != null)
                {
                    return EntityParser.EntityToObject(studentEntity);
                }
                else return null;
            }
        }
```

Martines Stefano
Magnin Adrien

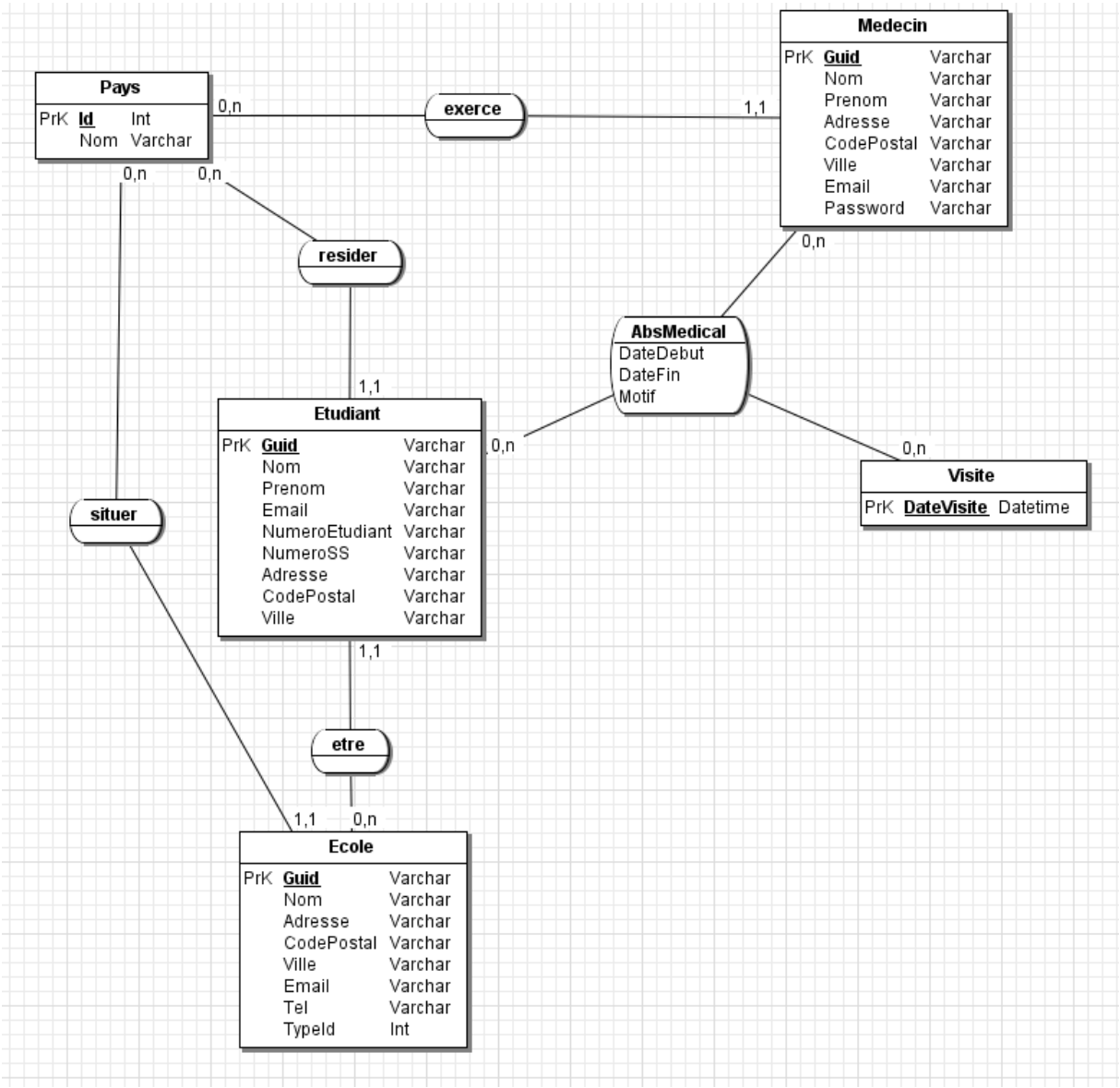
```
        public bool UpdateStudent(Student student)
        {
            using (rfidEntities db = new rfidEntities())
            {
                try
                {
                    var query = (from d in db.student where d.Guid == student.Guid select d).First();
                    query.Firstname = student.Firstname;
                    query.Lastname = student.Lastname;
                    query.Email = student.Email;
                    query.PostalCode = student.PostalCode;
                    query.City = student.City;
                    query.CountryId = student.CountryId;
                    query.Address = student.Address;
                    query.Phone = student.Phone;
                    query.SchoolGuid = student.SchoolGuid;
                    query.StudentId = student.StudentId;
                    db.SaveChanges();
                    return true;
                }
                catch (Exception)
                {
                    return false;
                }
            }
        }

        public bool RegisterStudent(Student student)
        {
            using (rfidEntities db = new rfidEntities())
            {
                db.student.Add(EntityParser.ObjectToEntity(student));
                int result = db.SaveChanges();
                return result > 0;
            }
        }
    }
```

Martines Stefano
Magnin Adrien

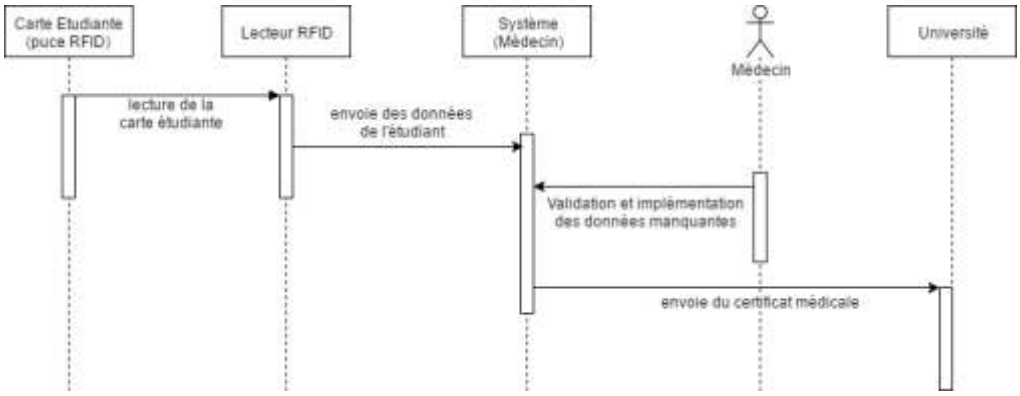
Annexes:

Schéma Conceptuel de Données



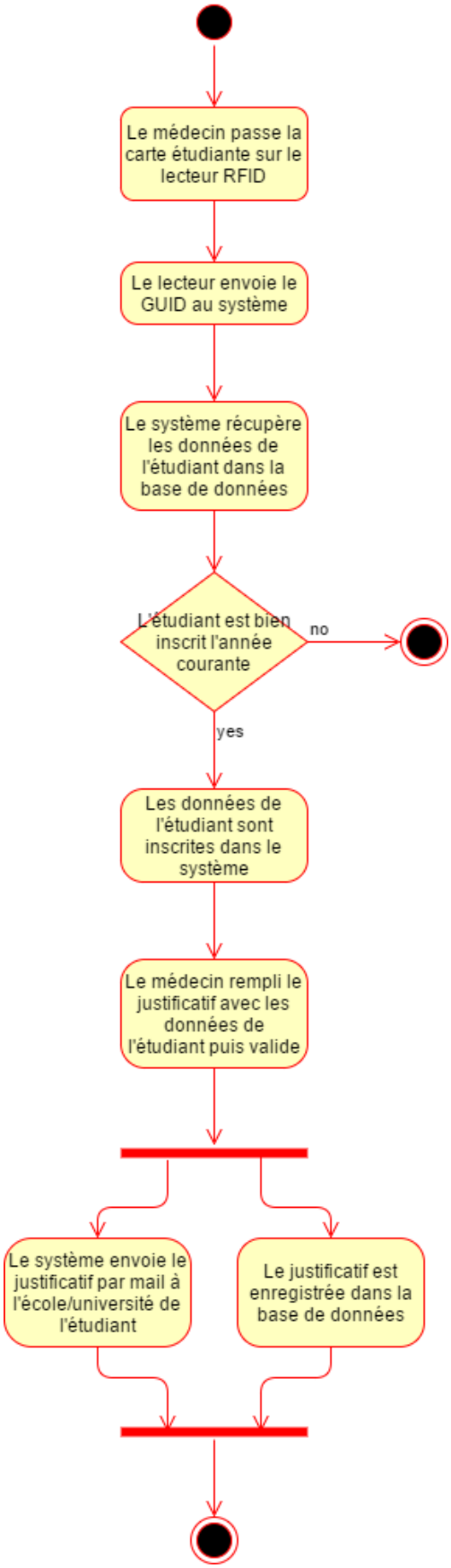
Martines Stefano
Magnin Adrien

Diagramme de Séquence



Martines Stefano
Magnin Adrien

Diagramme d'Activité



Martines Stefano
Magnin Adrien

Diagramme de Case d'Utilisation