



Office de la Formation Professionnelle et de la Promotion du Travail
Institut Spécialisé de Technologie Appliquée NTIC
Sidi Youssef Ben Ali Marrakech

Rapport de Stage

Projet RCD Manager

Concepteurs/Rédacteurs

Abdelhakim El hafid
Mohamed Elhadad
Houda Maârouf

Filière/Groupe

Techniques de Développement Informatique
Groupe TDI -2-A
Promotion 2016-2017

Société

RCD Service SARL
Guéliz Marrakech
Agence de location de véhicules

Dernière révision

22/05/2017 01:36

Table des matières

| | | |
|------|--|----|
| I. | Introduction | 3 |
| II. | Contexte et Etude | 3 |
| 1. | Société..... | 3 |
| a. | Présentation | 3 |
| b. | Services..... | 3 |
| c. | A propos du domaine | 3 |
| 2. | Problématique..... | 4 |
| 3. | Solution | 5 |
| III. | Analyse et Conception..... | 5 |
| 1. | Objectif..... | 5 |
| 2. | Fonctionnalités | 5 |
| 3. | Conception..... | 5 |
| a. | Composants du système..... | 5 |
| b. | Relations du système | 6 |
| c. | Modèle physique de données..... | 8 |
| IV. | Développement..... | 8 |
| 1. | Objectif..... | 8 |
| 2. | Contraintes..... | 8 |
| 3. | Outils..... | 9 |
| a. | Matériel..... | 9 |
| b. | Environnement de développement..... | 9 |
| c. | Langages..... | 10 |
| d. | Cadres applicatifs (Frameworks) | 12 |
| e. | Système de gestion de base de données (SGBD) | 13 |
| f. | Outils de modélisation | 14 |
| 4. | Conventions et nommage | 14 |
| a. | Noms des applications et des packages | 14 |
| b. | Base de données | 15 |
| c. | Application..... | 15 |
| 5. | Standards de programmation | 16 |
| 6. | Structure de l'application..... | 16 |
| a. | Interfaces..... | 16 |

| | |
|----------------------------|----|
| b. Structure..... | 17 |
| 7. Maquettes d'écran | 20 |
| V. Conclusion..... | 25 |

I. Introduction

Dans le cadre de notre projet de stage dans une agence de location de véhicules, nous nous intéressons au développement d'une application de bureau Windows pour répondre au problème de la non-informatisation qui touche à la société en manque d'outils informatiques.

Comme pour toute entreprise, l'informatisation peut conduire à d'importants gains de productivité, mais aussi à une amélioration de la qualité, éventuellement après une dématérialisation des documents. En fonction des modèles employés, l'informatisation peut conduire à certaines dérives productivistes, mais elle peut aussi être un bon outil de gestion de la qualité (notamment avec internet). Ainsi, l'informatisation peut avoir des conséquences variées en termes de développement durable.

Ce rapport décrit les étapes d'étude, d'analyse, de conception, et de développement de la solution d'informatisation proposée en mettant le focus sur les différents outils et notions de développement informatiques intéressants mis en œuvre.

II. Contexte et Etude

1. Société

a. Présentation

RCD Service (Rent a Car Douma Service) est une agence de **location de véhicules** (notamment les voitures) à courte durée créée en 2004 à Marrakech.

La société est constituée d'une **seule agence** et possède un seul parc comportant une dizaine de véhicules qui peuvent être réservés directement ou par appel téléphonique.

RCD Service représente un bon choix pour tous ceux ayant besoin d'un véhicule sans complexités pour les déplacements usuels pendant une brève durée et avec un prix raisonnable.

b. Services

Le service principal proposé par RCD Service offre à ses clients la possibilité de louer un automobile de leurs choix parmi ceux proposés pour une durée allant généralement d'une journée à quelques mois.

Des services supplémentaires sont aussi disponibles si désiré, notamment un conducteur, la livraison à l'emplacement souhaité, en plus des équipements utiles tels que le siège bébé et le GPS.

c. A propos du domaine

La location de voiture courte durée (LCD) ou location de véhicule est un service proposé par une société offrant aux clients la location des automobiles pour de courtes périodes pouvant aller de quelques heures à quelques semaines. Les sociétés de location sont souvent constituées de nombreuses agences locales permettant notamment aux clients de retourner leur véhicule à un endroit différent de celui de la prise en charge. Les agences de location sont situées pour la plupart dans les

terminaux des aéroports, à proximité immédiate des gares et dans les centres-villes et la réservation peut se faire en ligne par l'intermédiaire d'un site web.

La location de voitures répond en premier lieu à un besoin temporaire pour celles et ceux qui ne possèdent pas leur propre véhicule, qui ont besoin d'un véhicule de remplacement à la suite de l'immobilisation de leur automobile, qui recherchent un véhicule de complément après un voyage en train ou en avion.

Au-delà des voitures particulières, les véhicules utilitaires et les deux roues font partie intégrante de la location. En plus de la mise à disposition d'un véhicule, les agences proposent aussi de réserver des produits et services supplémentaires tels que : assurances, conducteur additionnel, systèmes de navigation GPS, sièges enfants, chaînes ou pneus neige, ou encore la mise à disposition du véhicule à l'hôtel.

2. Problématique

Pour gérer l'agence, les responsables utilisent des fiches et des registres en **papier** pour stocker les données relatives aux clients, aux véhicules, aux employés, et aux contrats de location. Ces documents sont fréquemment rangés sur le bureau du gérant, préservés dans un casier pour les plus anciens, ou même amenés ailleurs si jugés obsolètes à cause du manque d'espace pour les archiver.

Comme toute méthode traditionnelle de gestion, cette méthode a plusieurs inconvénients qui touchent à l'efficacité du travail. Ci-dessous, un récapitulatif de ces problèmes :

- L'enregistrement continu des données implique des **coûts** mensuels supplémentaires non négligeables pour les matériels épuisables tel que les papiers, cartons, registres, et stylos.
- La **recherche** d'informations dans les documents prend trop de **temps** et d'**effort** sans résultats satisfaisants.
- Tout éventuel excès des informations entrantes dû à plusieurs réservations en une même période conduit à un manque de **ressources** pour enregistrer les données et d'**espace** pour ranger les documents associés.

Pour remédier à ses problèmes, le commanditaire exprime son besoin pour une solution pouvant servir d'alternative à la méthode de gestion actuelle. Ci-dessous, un récapitulatif des principaux points à cibler :

- Une solution pour **réduire les coûts** sur les matériels et les outils utilisés dans l'enregistrement et la conservation des données.
- Une solution pour **faciliter la manipulation des données** et **réduire le temps et l'effort** requis pour la recherche d'informations.
- Une solution pour **gérer l'excès des informations** en préservant les ressources et en réduisant l'espace nécessaire pour le stockage des données.

Le gérant de l'agence dispose d'**un seul ordinateur** de bureau Windows sans **aucun accès à un réseau** qu'il souhaite utiliser pour profiter de la solution informatique qui sera proposée.

3. Solution

La solution informatique proposée pour cette problématique est une **application de bureau Windows** qui permet de cibler tous les points cités avant comme indiquée ci-dessous :

- La solution nécessite **un seul ordinateur** fonctionnel sans **aucun connectivité réseau requise** pour enregistrer et conserver les données. Les opérations de renouvellement des matériaux et des outils de travail ne sont donc pas très nécessaires tant que la machine fonctionne toujours ce qui réduit énormément les **coûts** mensuels.
- La solution **facilite** la manipulation des données et permet de rechercher les informations **rapidement** et **efficacement** sans aucun effort.
- La solution permet de stocker **énormément** de données sur le disque dur de l'ordinateur de façon très **organisé**, ce qui élimine tous les problèmes liés aux épuisement des ressources et au manque d'espace dans l'agence.

La solution sera développée pour fonctionner sur le poste de travail de l'agence en prenant en compte la **configuration technique** de l'ordinateur et les **contraintes** imposées par le comandataire.

III. Analyse et Conception

1. Objectif

L'objectif est de fournir un système destiné à gérer l'agence de location de véhicules **RCD Service**. Ce système constitue une **application de bureau Windows (Windows Desktop Application)** installée sur le poste de travail de l'agence.

2. Fonctionnalités

L'application permet à l'employé responsable de l'agence (gérant ou assistant) de **gérer les composants du système** (Contrats de location, réservations, véhicules, clients, et employés), de suivre l'état des véhicules (Inspection et Maintenance), et enfin de **générer des rapports** de location imprimables.

3. Conception

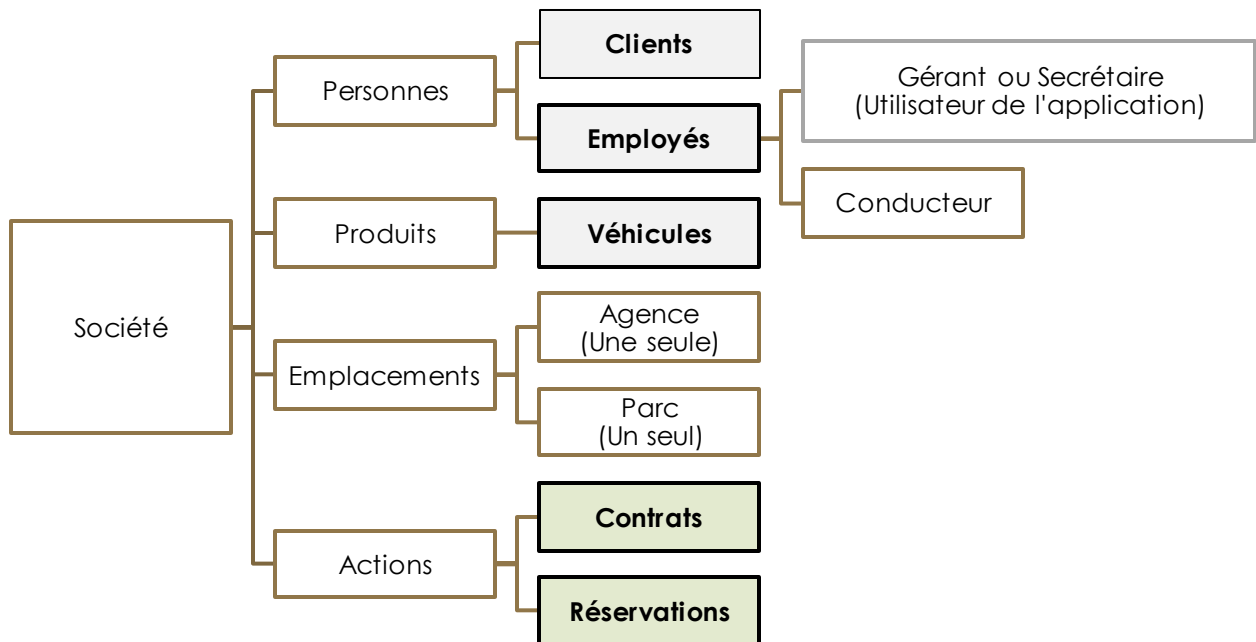
Les méthodes de conception sont utilisées afin d'améliorer la qualité de la conception finale.

Différentes méthodes individuelles ont été utilisées pour analyser le système, ces méthodes sont inspiré de la méthode de conception **MERISE** qui permet de mettre en place le Modèle Conceptuel de Données (MCD) et le Modèle Logique de Données (MLD) pour aboutir enfin au script de génération de la base de données.

Les schémas, plans, et diagrammes correspondants aux modèles d'analyse et de conception utilisés ont été initialement créé à l'aide de l'outil de cartes heuristiques X-Mind.

a. Composants du système

Le schéma suivant représente l'ensemble des composants du système de la société :



Depuis le schéma précédent, on peut déduire les entités qui vont pouvoir être utilisées pour modéliser le système, comme ci-dessous :

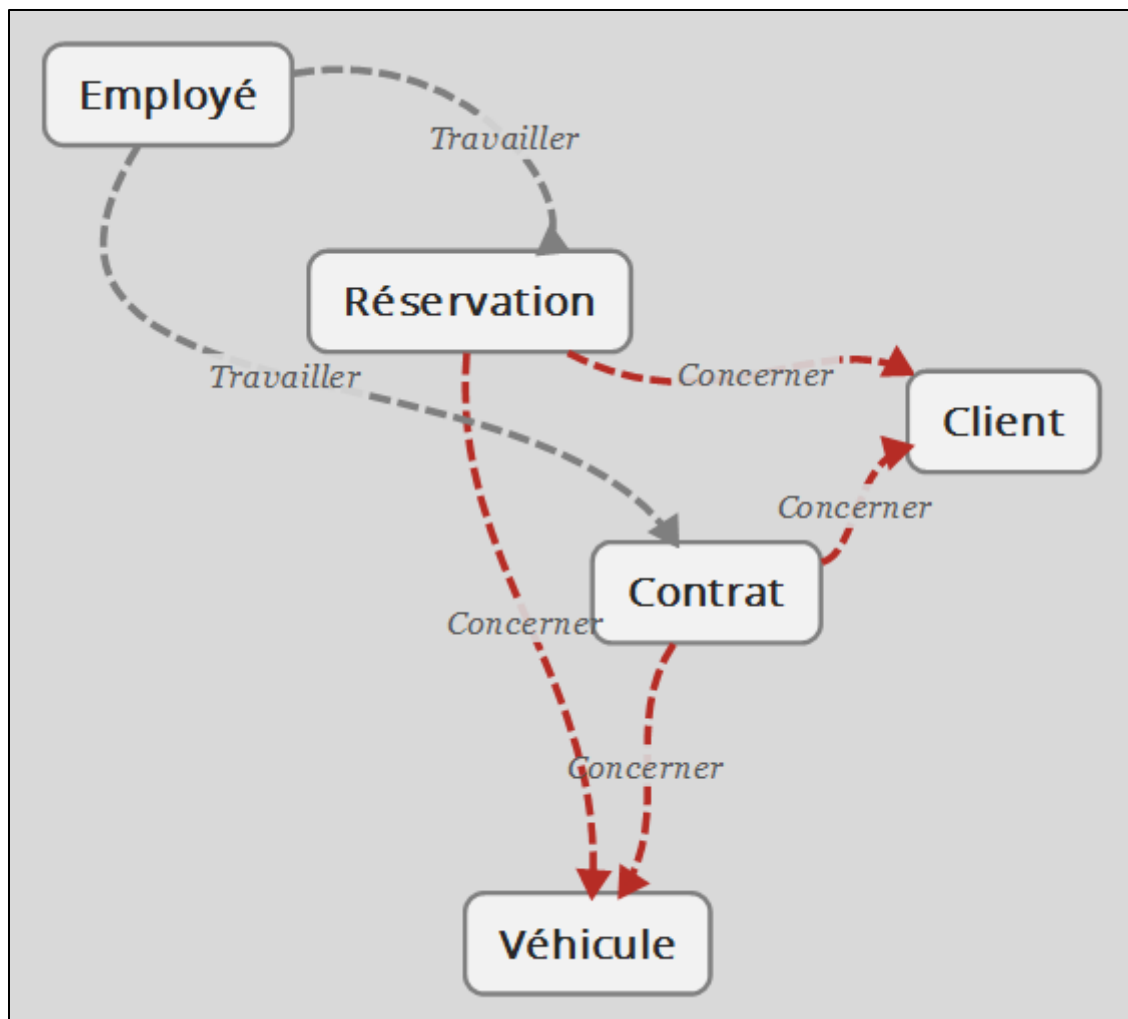
| Client | Employé | Véhicule | Contrat | Réservation |
|--------|---------|----------|---------|-------------|
| | | | | |

b. Relations du système

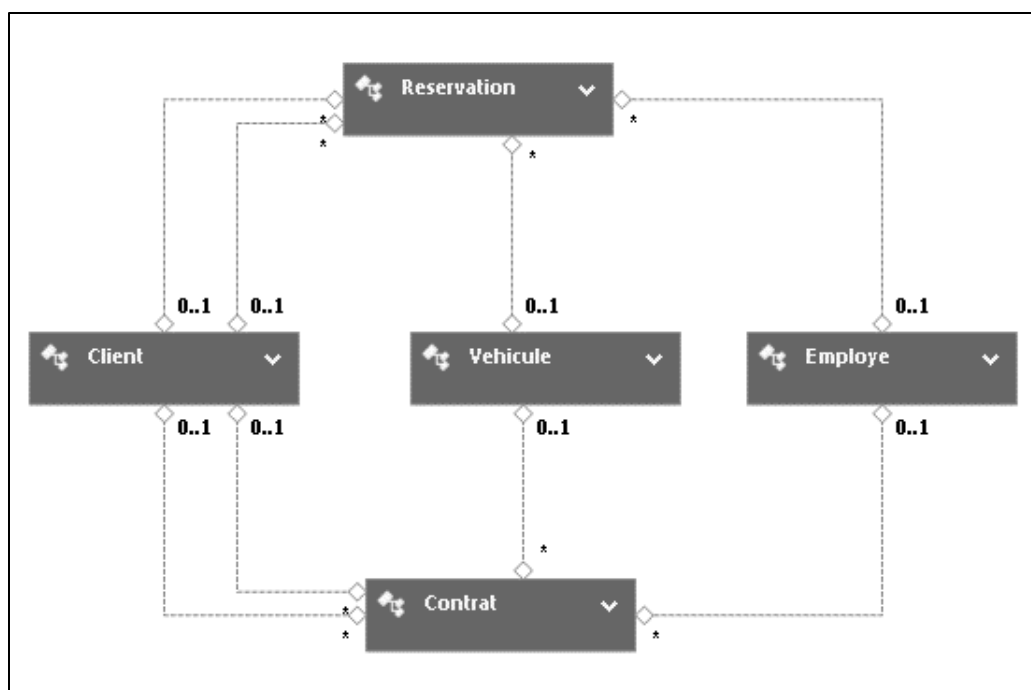
Les relations du système sont issues des règles de gestion qui décrivent la nature des relations entre les entités du système d'information. L'ensemble de ces règles permet de définir un système correspondant au problème étudié précisément adaptée aux besoins du commanditaire. Ces règles de gestion sont utilisées directement dans le modèle physique de donnée. Chaque règle correspond à une relation entre deux (ou plusieurs) entités. Ci-dessous un récapitulatif des règles de gestion imposées par le système de la société :

- Un **client** peut signer un ou plusieurs **contrats** ou **réservations**.
- Un **contrat** ou **réservation** correspond à un seul **client** et un seul **véhicule**.
- Dans un **contrat** ou **réservation**, un **client** peut avoir un seul **compagnon (client)** pouvant aussi conduire le véhicule à ses côtés.
- Dans un **contrat** ou **réservation**, un **client** peut avoir un seul **conducteur (employé)** avec le véhicule.

Le schéma suivant représente l'ensemble des relations du système :

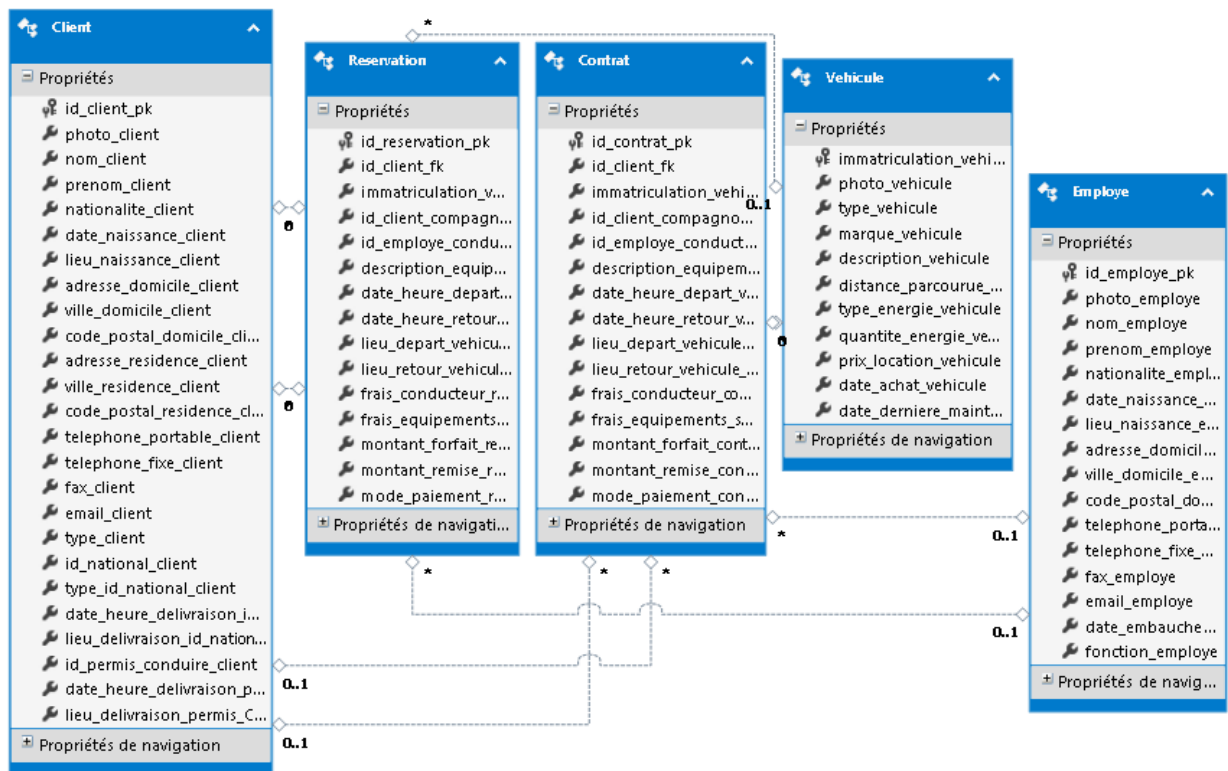


Le schéma suivant représente l'ensemble des relations avec cardinalités :



c. Modèle physique de données

Le schéma suivant représente l'ensemble des tables construites à partir des composants du système et leurs relations :



IV. Développement

1. Objectif

L'objectif est de créer une **application de bureau Windows** permettant de **gérer le système** de l'agence de location de véhicules RCD Service à partir du **modèle de données** précédemment conçu.

2. Contraintes

Plusieurs contraintes provenant de différentes sources sont à prendre en compte dans la phase de conception du système. Ci-dessous, un récapitulatif des contraintes imposées par le commanditaire :

- L'application doit être destinée à fonctionner sur **un seul poste de travail** (Bureau ou portable) sous **Windows** (précisément Windows 7).
- Toutes les données de l'application doivent être stockées dans la **même location que l'application** (Base de données locale) et ne doivent être **accessible par ailleurs** (Pas de réseau, un seul poste).
- L'interface graphique de l'application doit être **adaptée** pour différentes résolutions d'écrans de moniteurs, et être facile à utiliser sur un écran **tactile**.

3. Outils

a. Matériel

Le matériel de développement utilisé est une machine équipée de **Windows 7 Professionnel ou Intégrale** et d'une quantité suffisante de mémoire vive (minimum **1Go** pour une qualité de développement acceptable, en vue des nombreux services devant être exécutés en cours du développement).

La gestion du développement collaboratif (en équipe) a été fait manuellement, aucun outil n'a pas pu être utilisé (les outils de développement collaboratif nécessitent quelques matériels spéciaux en possession des différents membres de l'équipe de développement pour assurer leurs fonctionnement).

Les trois membres de l'équipe de développement exécutent leurs versions de l'application sur leurs propres machines. La base de données SQL Server est transférée d'une machine à l'autre à travers un **fichier de base de données à importer**. Une fusion (gestion manuelle des éventuels conflits) des trois versions de l'application (chaque version fournit par un membre de l'équipe) et de la base de données mise à jour constitue une version finale de l'application. **Une seule machine** peut donc être suffisante pour le développement du projet.

b. Environnement de développement

L'application est articulée sur la plate-forme **.NET** de Microsoft, l'environnement de développement intégrée **Visual Studio 2015** est mis à disposition de l'équipe de développement.

Microsoft Visual Studio est un **environnement de développement intégré (IDE)** de Microsoft. Il est utilisé pour développer des programmes informatiques pour **Microsoft Windows**, ainsi que des **sites Web**, des **applications Web**, des **services Web** et des **applications mobiles**. Visual Studio utilise des plates-formes de développement de logiciels Microsoft telles que **Windows API**, **Windows Forms**, **Windows Presentation Foundation**, **Windows Store** et **Microsoft Silverlight**. Il peut produire à la fois le **code natif** et le **code managé**.

Visual Studio inclut un **éditeur de code** prenant en charge **IntelliSense (le composant d'achèvement du code)** ainsi que le **refactorisation** de code. Le **débogueur intégré** fonctionne à la fois comme un débogueur de niveau source et un débogueur au niveau machine. Les autres outils intégrés incluent un **profilier de code**, un **concepteur de formulaires** pour créer des **applications GUI**, un **concepteur de sites Web**, un **concepteur de classe** et un **concepteur de schéma de base de données**. Il accepte les **plug-ins** qui améliorent la fonctionnalité à presque tous les niveaux, y compris l'ajout de support pour les systèmes de contrôle source (comme Subversion) et l'ajout de nouveaux outils tels que les éditeurs et les concepteurs visuels pour les langues ou les jeux d'outils spécifiques au domaine pour d'autres aspects du **cycle de développement** (Comme le client **Team Foundation Server**: Team Explorer).

Visual Studio prend en charge **36 langues de programmation différentes** et permet à l'éditeur de code et au débogueur de supporter (à des degrés divers) presque **n'importe quel langage de programmation**, à condition qu'un service spécifique à la langue existe. Les langages intégrés incluent **C**, **C ++** et **C ++ / CLI** (via Visual C ++), **VB.NET** (via Visual Basic .NET), **C #** (via Visual C #), **F #** (à partir de Visual Studio 2010) Et

TypeScript (à partir de Visual Studio 2013 Update 2). Le support d'autres langues telles que **Python**, **Ruby**, **Node.js** et **MUMPS**, entre autres, est disponible via les services de langue installés séparément. Il prend également en charge **XML / XSLT**, **HTML / XHTML**, **JavaScript** et **CSS**. **Java** (et **J#**) ont été pris en charge par le passé.

Microsoft fournit une version gratuite de Visual Studio appelée l'édition communautaire qui prend en charge les plug-ins et est disponible sans frais.

c. Langages

Voici la liste des principaux langages utilisés dans le projet RCD :

- **SQL** pour les scripts de création et gestion de la base de données (création de tables et mise à jour des enregistrements) ainsi que pour les requêtes spécifiques de l'application.
 - **Structured Query Language** est un **langage de requêtes** structurées conçu pour gérer les données Détenue dans un **système de gestion de base de données relationnelle** (RDBMS), ou pour le traitement de flux dans un **système de gestion de flux de données relationnel** (RDSMS). Initialement basé sur l'**algèbre relationnelle** et le **calcul relationnel de tuple**, SQL consiste en un **langage de définition de données**, un **langage de manipulation de données** et un **langage de contrôle de données**. La portée de SQL comprend l'**insertion de données**, la **requête**, la **mise à jour** et la **suppression**, la **création** et la **modification** de schéma et le **contrôle d'accès** aux données. Bien que SQL soit souvent décrit comme étant, dans une large mesure, un **langage déclaratif** (4GL), il comprend également des éléments de **procédure**. SQL a été l'une des premières langues commerciales pour le **modèle relationnel** d'Edgar F. Codd, tel qu'il est décrit dans son article influent intitulé « Un modèle relationnel de données pour les grandes banques de données partagées ». Malgré l'absence de tout le respect du **modèle relationnel** tel que décrit Par Codd, il est devenu la langue de base de données **la plus utilisée**. SQL est devenu une norme de l'**American National Standards Institute (ANSI)** en 1986 et de l'**Organisation internationale de normalisation (ISO)** en 1987. Depuis lors, la norme a été révisée pour inclure un plus grand ensemble de fonctionnalités. Malgré l'existence de telles normes, la plupart des codes SQL **ne sont pas complètement portables** parmi les **différents** systèmes de base de données **sans ajustements**.
- **Visual C#** pour le développement de l'application.
 - **Microsoft Visual C #** est l'implémentation Microsoft de la spécification C#, incluse dans la suite de produits **Microsoft Visual Studio**. Il est basé sur la spécification ECMA/ISO de la langue C#, que Microsoft a également créée. Bien que des implémentations multiples de la spécification existent, Visual C# est de loin le plus couramment utilisé. Dans la plupart des contextes, une référence non qualifiée à "C#" signifie "Visual C#".
 - **C#** (C Sharp) est un **langage de programmation multi-paradigme** englobant des disciplines de programmation forte, impérative, déclarative, fonctionnelle, générique, orientée objet (basée sur la classe) et orientée vers les composants. Il a été développé par **Microsoft**

dans son initiative **.NET** et plus tard approuvé en **standard** par Ecma (ECMA-334) et ISO (ISO / CEI 23270: 2006). C# est l'une des langages de programmation conçus pour l'infrastructure de **langage commun**. C# est un langage de programmation **général** et **orienté objet**. Son équipe de développement est dirigée par Anders Hejlsberg. La version la plus récente est C# 7.0 qui a été publié en 2017 avec Visual Studio 2017.

D'autres langages ont été utilisé indirectement à travers les générateurs de code et les assistants impliquées dans les différents processus de développement comme :

- **XML** pour les fichiers de configuration et les paramètres de l'utilisateur.
 - **Extensible Markup Language (XML)** est un **langage de balisage** qui définit un ensemble de règles pour coder des documents dans un format à la fois **humain** et lisible par **machine**. La spécification XML 1.0 du **W3C** et plusieurs autres spécifications connexes - toutes d'entre elles des **standards ouverts libres** - définissent XML. Les objectifs de conception de XML mettent l'accent sur la **simplicité**, la **généralité** et la **convivialité** sur Internet. Il s'agit d'un format de données **textuelles** avec un support fort via **Unicode** pour différentes **langues humaines**. Bien que la conception de XML se concentre sur les **documents**, la langue est largement utilisée pour la représentation de **structures de données arbitraires** telles que celles utilisées dans les **services Web**. Plusieurs **systèmes de schéma** existent pour aider à la définition des langages basés sur XML, tandis que les programmeurs ont développé de nombreuses **interfaces de programmation d'applications (API)** pour faciliter le **traitement** des données XML.
- **LINQ** (langage à requêtes intégrée à Visual C#, similaire à SQL) pour les requêtes ciblant les classes générées par l'assistant d'échafaudage de Dev Express depuis les modèles d'Entity Framework.
 - **Language Integrated Query (Requête intégrée au langage**, aussi connu sous le nom de LINQ) est un composant du cadre **.NET** de Microsoft qui ajoute les capacités d'interrogation sur les données aux langages **.NET** dans une syntaxe proche de celle de **SQL**. LINQ définit un ensemble d'**opérateurs de requêtes** qui peuvent être utilisés pour **l'exécution des requêtes**, **filtrer** et **projeter** des données dans les **collections**, dans les **classes énumérables**, dans les **structures XML**, dans les **bases de données relationnelles** et dans les **sources de données Tierces**. Bien qu'il permette d'effectuer des requêtes sur **n'importe quelle source de données**, il est nécessaire de les saisir dans les **objets**. Par conséquent, si la source de données ne stocke pas **nativement** les données en tant qu'objets, les données doivent être **mappées** sur le domaine correspondant. Les requêtes effectuées à l'aide d'opérateurs LINQ sont exécutées soit directement par le moteur de traitement de LINQ, soit par un mécanisme d'extension géré par les fournisseurs LINQ. Ceux-ci implémentent leur propre **moteur de traitement de requête**, ou convertissent les données dans un format approprié à un système de stockage différent (tel qu'une

base de données par exemple). Le résultat de la requête est renvoyé sous forme d'une **collection d'objets** qui peuvent être énumérée.

d. Cadres applicatifs (Frameworks)

Voici la liste des principaux cadres applicatifs utilisés dans le projet RCD :

- **.NET** pour le développement global de l'application.
 - **.NET Framework** est un cadre applicatif de logiciels développé par Microsoft qui fonctionne principalement sur **Microsoft Windows**. Il comprend une grande bibliothèque de classes nommée **Framework Class Library (FCL)** et fournit une **interopérabilité de langue** (chaque langue peut utiliser un code écrit dans d'autres langues) dans plusieurs langues de programmation. Les programmes écrits pour .NET Framework s'exécutent dans un **environnement logiciel** (contrairement à un environnement matériel) nommé **Common Language Runtime (CLR)**, une **machine virtuelle** d'application qui fournit des services tels que **la sécurité, la gestion de la mémoire et la gestion des exceptions**. (En tant que tel, le code informatique écrit à l'aide de .NET Framework s'appelle **code managé**) **FCL** et **CLR** constituent ensemble .NET Framework.
 - **FCL** fournit **l'interface utilisateur, l'accès aux données, la connectivité de la base de données, la cryptographie, le développement d'applications Web, les algorithmes numériques et les communications réseau**. Les programmeurs produisent des logiciels en combinant leur code source avec .NET Framework et d'autres bibliothèques. Le cadre est destiné à être utilisé par la plupart des nouvelles applications créées pour **la plateforme Windows**. Microsoft produit également un environnement de développement intégré en grande partie pour le logiciel .NET appelé **Visual Studio**.
- **DevExpress** pour les interfaces graphiques et les outils d'échafaudage.
 - **DevExpress Universal** est une suite de **composants .NET** de haute qualité pour **Visual Studio**. Elle propose **des contrôles de présentation** complets, **des outils de productivité IDE**, des **Framework Business** et **des systèmes de reporting**, ainsi que **des frameworks mobiles HTML-JS** de haute performance pour les développeurs ciblant **iOS, Android et Windows Phone**. Que ce soit à l'aide de **WPF, ASP.NET, WinForms, HTML5** ou **Windows 10**, les outils de DevExpress aident à offrir la meilleure expérience de **développement .NET** possible.
- **ADO.NET** pour l'accès aux données.
 - **ADO.NET** est une technologie d'**accès aux données** de Microsoft **.NET Framework** qui fournit une communication entre les **systèmes relationnels** et **non relationnels** à travers un ensemble commun de composants. ADO.NET est un ensemble de composants logiciels informatiques que les programmeurs peuvent utiliser pour **accéder** aux données et aux services de données à partir de **la base de données**. C'est une partie de la bibliothèque de classes de base qui est incluse dans Microsoft .NET Framework. Il est couramment utilisé par les programmeurs pour accéder et modifier les données stockées dans **les**

systèmes de bases de données relationnelles, même si elles peuvent également accéder à des données dans **des sources non relationnelles**. ADO.NET est parfois considéré comme une évolution de la technologie **ActiveX Data Objects (ADO)**, mais a été tellement modifié qu'il peut être considéré comme un tout nouveau produit.

- **Entity Framework** pour le modèle de données.
 - **Entity Framework (EF)** est un outil de **mapping objet-relationnel (ORM)** open source pour **ADO.NET**, une partie de **.NET Framework**. Il s'agit d'un ensemble de technologies dans ADO.NET qui prend en charge le développement d'applications logicielles orientées données. Les architectes et les développeurs d'**applications orientées données** ont généralement lutté avec la nécessité d'atteindre deux objectifs très différents. Entity Framework permet aux développeurs de travailler avec des données sous la forme d'**objets** et de **propriétés** spécifiques au domaine, tels que les clients et les adresses clients, sans avoir à se préoccuper des **tables** et des **colonnes** sous-jacentes où ces données sont stockées. Avec Entity Framework, les développeurs peuvent travailler à un niveau d'abstraction plus élevé lorsqu'ils traitent des données et peuvent créer et gérer des applications orientées données avec moins de code que dans les applications traditionnelles. Depuis la version 6, Entity Framework est séparé du framework .NET.
 - **Le mappage objet-relationnel (ORM)** est une technique de programmation pour **convertir des données** entre des systèmes de type **incompatibles** dans des langages de programmation **orientés objet**. Cela crée en fait une « base de données d'objets virtuels » pouvant être utilisée dans le langage de programmation. Il existe à la fois des packages gratuits et commerciaux disponibles qui effectuent **un mappage objet-relationnel**, bien que certains programmeurs choisissent de construire leurs propres outils ORM.

e. Système de gestion de base de données (SGBD)

Le projet RCD utilise une base de données locale **SQL Server Express** intégrée à **Visual Studio**.

Microsoft SQL Server Express est une version de **SQL Server** libre de téléchargement, de distribution et d'utilisation. Il comprend une base de données spécifiquement destinée aux **applications intégrées** et à **petite échelle**. Le produit trouve ses racines dans le produit **Microsoft Database Engine (MSDE)**, qui a été livré avec **SQL Server 2000**. La marque « Express » a été utilisée depuis la publication de **SQL Server 2005**.

Microsoft SQL Server est un **système de gestion de base de données relationnel** développé par Microsoft. En tant que **serveur de base de données**, il s'agit d'un produit logiciel dont la fonction principale consiste à **stocker** et à **récupérer** des données comme demandé par d'autres applications logicielles, qui peuvent être exécutées sur un **même ordinateur** ou sur un **autre ordinateur** sur un **réseau** (y compris Internet).

Microsoft commercialise au moins une douzaine d'éditions différentes de Microsoft SQL Server, destinées à différents publics et à des charges de travail, allant des **petites**

applications à une seule machine à de **grandes applications** face à Internet avec de nombreux utilisateurs simultanés.

La base de données SQL Server du projet RCD est manipulée et testée grâce à l'outil **dbForge Studio For SQL Server** (lors de la phase de développement, un ensemble de données tests est utilisé afin d'avoir un support convenable pour les différents services à créer).

Devart DbForge Studio For SQL Server est un **environnement de développement intégré (IDE)** développée par Devart pour **la gestion, l'administration, le développement, le reporting et l'analyse** de données SQL Server. L'outil aide les développeurs SQL à effectuer des modifications complexes dans les bases de données, de contrôler les changements de versions de base de données dans les logiciels de gestion de versions populaires, et d'accélérer les tâches de routine. Il s'agit d'une puissante alternatif à **SQL Server Management Studio (SSMS)**, le gestionnaire SQL Server par défaut de Microsoft.

f. Outils de modélisation

L'**outil de cartes heuristiques X-Mind** est utilisé pour la réalisation des schémas et diagrammes correspondants aux modèles. Le script de création de la base de donnée et des tables associées a été écrit manuellement à partir des modèles réalisés.

XMind est un **logiciel de mappage et de brainstorming**, développé par XMind Ltd. Outre les éléments de **gestion**, le logiciel peut **capturer des idées, clarifier la réflexion, gérer des informations complexes** et promouvoir la **collaboration d'équipe** pour une **productivité** plus élevée. En avril 2013, XMind a été sélectionné comme le **logiciel de cartographie mentale** le plus populaire sur Lifehacker.

X-Mind prend en charge **les cartes d'esprit, les diagrammes de la cape, les diagrammes d'arbres, les organigrammes, les feuilles de calcul**, etc. Normalement, il est utilisé pour **la gestion des connaissances, les procès-verbaux, la gestion des tâches** et **GTD**. Pendant ce temps, XMind peut lire les fichiers **FreeMind** et **MindManager** et enregistrer dans **Evernote**. Pour **XMind Pro ou Plus**, il peut même **exporter** les cartes mentales dans les documents **Microsoft Word, PowerPoint, Excel, PDF, FreeMind** et **Mindjet MindManager**.

4. Conventions et nommage

Ces conventions concernent les répertoires et dossiers, des entités spécifiques dans les applications du système RCD (classes, variables, packages, entités de fichiers de configuration) et dans la base de données (tables, champs).

a. Noms des applications et des packages

Le système complet s'appelle « RCD ». Chaque composant du projet possède un nom spécifique qui permet de l'identifier dans le système (**RCD Manager** pour l'application déployée, **RCD Solution** pour le code, et **RCD Database** pour la base de données). Les packages utilisés dans les sources sont définis comme suit :

- La racine de tout le système est : RCD
- L'exécutable final de l'application déployée est située dans le dossier : « RCD Manager »
- Le code source de l'application est située dans le dossier : « RCD Solution »

- Le script de la base de données et des données de test sont situées dans le dossier : « RCD Database »

b. Base de données

Les **tables de la base de données** sont issues de deux catégories de données dans le MCD : les entités (employés, clients, etc.) et les associations (rattachement d'un contrat à un client, mise à disposition d'un véhicule par un employé, etc.). Chaque **entité** possède un libellé lisible qui permet de la distinguer clairement (table Client au lieu de CLI par exemple). Ce libellé peut contenir des chiffres, mais ne peut pas contenir d'articles dans le nom (ex : LE_CLIENT), ni de verbes (VEHICULE_RATTACHER_A).

Les **associations** reliant toujours deux entités au minimum, le libellé des tables correspondantes représente clairement la relation entre les deux entités, ce qui donne par exemple : Contrat pour la table correspondant à l'association rattachant un véhicule à un client.

Concernant **les champs de la base de données**, les noms des colonnes ont les mêmes règles que les libellés des tables. Le nom de chaque colonne d'une table commence par le nom du champ suivi du nom de la table suivi d'un suffixe si la colonne est une clé primaire, étrangère, ou les deux (PK, FK, FK_PK). Ceci permet d'identifier très clairement les différentes colonnes.

Concernant les **requêtes SQL** présentes dans les fichiers sources, elles doivent être écrites de la façon suivante : lettres majuscules pour les mots SQL (ex. : SELECT, UPDATE, CREATE) et lettres en minuscules ou capitales pour les noms des objets sur lequel porte la requête (noms des tables, champs, variables, etc.). Avant chaque **mot SQL**, il est souhaitable d'avoir un retour à la ligne et d'aligner les lignes de la requête (avec des espaces et non des tabulations qui ne sont pas portables pour la mise en page). De même, les conditions des sous-requêtes doivent être décalées par rapport à la requête principale.

c. Application

Au sein de l'application, il convient de nommer chaque groupe de composants de façon prédéfinie. En effet, chaque ensemble de composants liés est stockée dans un ou plusieurs fichiers ou dossiers différents :

- **Le projet de l'application** est contenu dans le dossier *RCD Project*.
- **La base de données** est stockée dans le fichier : *RCD_Database.mdf*.
- **Le modèle de données** est stocké dans le fichier : *RCD_Model.edmx*
- Couches **MVVM**
 - **La couche modèle de données** est contenue dans le dossier *RCD_ConnectionDataModel*.
 - **La couche modèle de vue** est contenue dans le dossier *ViewModels*. Ce dossier contient des sous-dossiers nommés d'après les différentes tables et vues issues de la source de données.
 - **La couche vue** est contenue dans le dossier *Views*. Ce dossier contient également des sous-dossiers nommés d'après les différentes tables et

vues issues de la source de données. La vue principale se trouve à la racine du dossier.

- Les **classes d'accès aux données (Data Access Interface)** sont contenues dans le fichier : *DAI.cs*.
- Les **classes d'accès aux services (Application Programming Interface)** sont contenues dans le fichier : *API.cs*.
- Le **rapport** est associé au fichier : *Contrat_Report.cs*
- Le **tableau de bord** est associé fichier : *Dashboard1.cs*
- Formulaires indépendants
 - Le **formulaire de démarrage** est associé au fichier *Form_Main.cs*.
 - Le **formulaire de paramètres** est associé au fichier *Form_Settings.cs*.
 - Le **formulaire de notification** est associé au fichier *Form_Alert.cs*.

5. Standards de programmation

L'équipe de développement suit un ensemble de conventions de codage qui permettent une homogénéisation des sources :

- Les **commentaires** doivent être rédigés pour chaque classe et chaque méthode étrangère aux outils prédéfinis en respectant la norme MSDN, afin de permettre à toute personne entrant dans le projet de reconnaître l'utilité, les entrées et les sorties de ces entités.
- Les **conventions de codage C#** utilisées sont celles recommandées par le MSDN (Guide de programmation C#).
- Les **scripts SQL** sont écrits suivant la norme de formatage unifiée du code SQL.

6. Structure de l'application

a. Interfaces

- **Interfaces utilisateur**
 - L'**interface utilisateur de l'application** est une interface DESKTOP (via le système d'exploitation Windows) contenant des formulaires et des tableaux de données regroupés par fonctions, le tout agencé par une navigation par onglets. Cette interface DESKTOP permet l'affichage et la saisie de données. Cette application ne requiert pas d'authentification de la part de l'utilisateur, son accès reste donc confirmé par Windows.
- **Interfaces logicielles**
 - **Interface avec la base de données locale** : l'application installée sur le DESKTOP doit interagir avec la base de données embarquée qui l'accompagne (SQL Server Express, Base de données locale intégrée), grâce à un pilote dédié.
 - Aucune base de données externe n'est prévue pour interagir avec le système. La seule base de données, mentionnée précédemment, fait partie du système à concevoir.
- **Interfaces de communication**

- Aucune interface de communication à distance n'est prévue pour interagir avec le système. L'application fonctionnent localement et ne requiert aucune connexion à un réseau pour fonctionner.

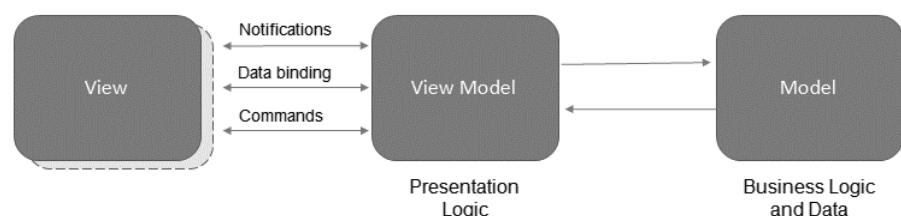
b. Structure

L'arborescence du projet est composée de plusieurs nœuds comme indiquée ci-dessous :

- **Propriétés**
 - Le nœud Propriétés représente des paramètres de configuration qui s'appliquent à l'ensemble du projet et qui sont stockés dans le fichier .csproj, dans le dossier de la solution. Ces paramètres incluent notamment des options de compilation et des paramètres de sécurité et de déploiement.
- **Références**
 - Une référence identifie un fichier binaire dont l'application a besoin pour s'exécuter. En général, une référence identifie un fichier .dll tel que l'un des fichiers de bibliothèque de classes .NET Framework. Il peut également référencer un assembly .NET (appelé un déflecteur) qui permet à l'application d'appeler des méthodes sur un objet COM ou une DLL Win32 native. Si le programme crée une instance d'une classe qui est définie dans un autre assembly, il faut ajouter une référence à ce fichier dans le projet avant de le compiler. Par défaut, tous les projets C# incluent une référence à mscorlib.dll, qui contient les classes .NET Framework principales.
- **Ressources**
 - Une ressource représente des données qui sont incluses avec l'application mais qui peuvent être stockées de façon à ce qu'elles puissent être modifiées indépendamment de l'autre code source. Par exemple, on peut stocker toutes les chaînes en tant que ressources au lieu de les coder en dur dans le code source. On peut traduire ensuite les chaînes dans différents langages ultérieurement et les ajouter au dossier d'application expédié aux clients sans devoir recompiler l'assembly. Les cinq types de ressources définis par Visual C# sont les suivants : chaînes, images, icônes, audio et fichiers. On peut ajouter, supprimer ou modifier des ressources à l'aide du Concepteur de ressources.
- **Modèle de données Entity Framework (fichier .edmx)**
 - Le fichier .edmx est un fichier XML qui définit un modèle conceptuel, un modèle de stockage et le mappage entre ces modèles. Un fichier .edmx contient également des informations utilisées par ADO.NET Entity Data Model Designer (Entity Designer) pour restituer graphiquement un modèle. Le fichier .edmx est créé avec l'Assistant EDM. Les modifications sont apportées au fichier .edmx lors de l'utilisation du Concepteur d'entités pour modifier le modèle et lors de l'utilisation de l'Assistant de mise à jour pour mettre à jour le modèle selon les modifications apportées à la base de données sous-jacente.

- **Couches de l'architecture MVVM générée par l'assistant d'échafaudage (Scaffolding Wizard) de DevExpress**

- **L'assistant d'échafaudage** est un outil design-time qui génère une application entièrement fonctionnelle prête à l'emploi basée sur la source de données utilisée. L'assistant est accessible à partir de la galerie de modèles DevExpress en tant que modèle séparé. Une application générée est construite selon les concepts MVVM et comprend trois types d'écrans d'application (modules) :
 - **Un écran d'application de démarrage.** Cet écran contient des éléments d'interface utilisateur de navigation qui mènent les utilisateurs finaux aux vues de collection (collection views).
 - **Vues de collection.** Ces écrans affichent toutes les entités de données à partir de tables de base de données connexes (clients, employés, contrats, etc.). Si les utilisateurs finaux sont autorisés à modifier ces enregistrements ou à en ajouter de nouveaux, ces modifications sont effectuées dans des vues détail (detail views).
 - **Vues détail** ou vues édition (edit views). Ces modules affichent un enregistrement de données spécifique à partir d'un pool (informations de client, détails de contrat, etc.).
- **DevExpress MVVM Framework** permet d'utiliser le modèle de conception **Model-View-ViewModel** dans les applications WinForms. Conçu pour le développement WPF, MVVM est un modèle de conception architecturale qui sépare une application en **trois couches** qui composent le titre du motif :
 - **Modèle (Model)**, définit les données et la logique métier.
 - **Vue (View)**, spécifie l'interface utilisateur, y compris tous les éléments visuels (boutons, étiquettes, éditeurs, etc.) liés aux propriétés et aux commandes dans la **Vue-Modèle**.
 - **Vue-Modèle (ViewModel)**, connecte le **modèle** et la **vue**. Cette couche est une abstraction de la **Vue** qui expose les propriétés publiques et les commandes utilisées pour lier les données aux éléments de l'interface graphique utilisateur et gérer les données. Le schéma ci-dessous illustre ces couches et leurs modes de communication :

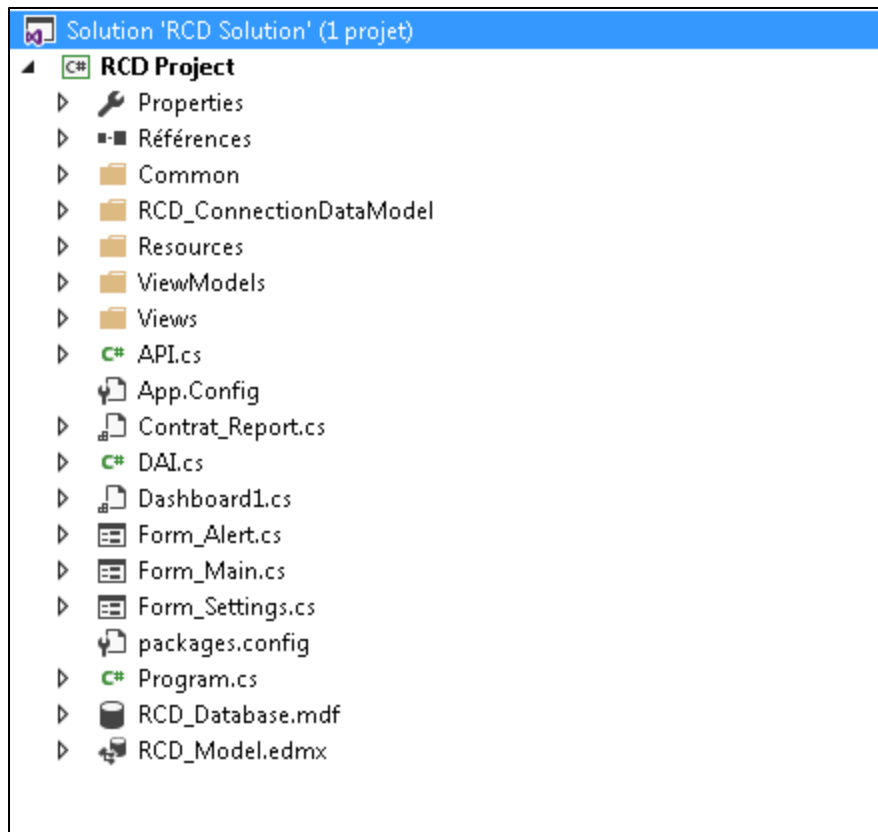


Cette séparation accorde de multiples avantages, comme un processus de développement plus indépendant pour les développeurs et les concepteurs, des tests de code plus faciles ou une conception plus simple de l'interface utilisateur.

- **La couche modèle de données** est contenue dans le dossier *RCD_ConnectionDataModel*.

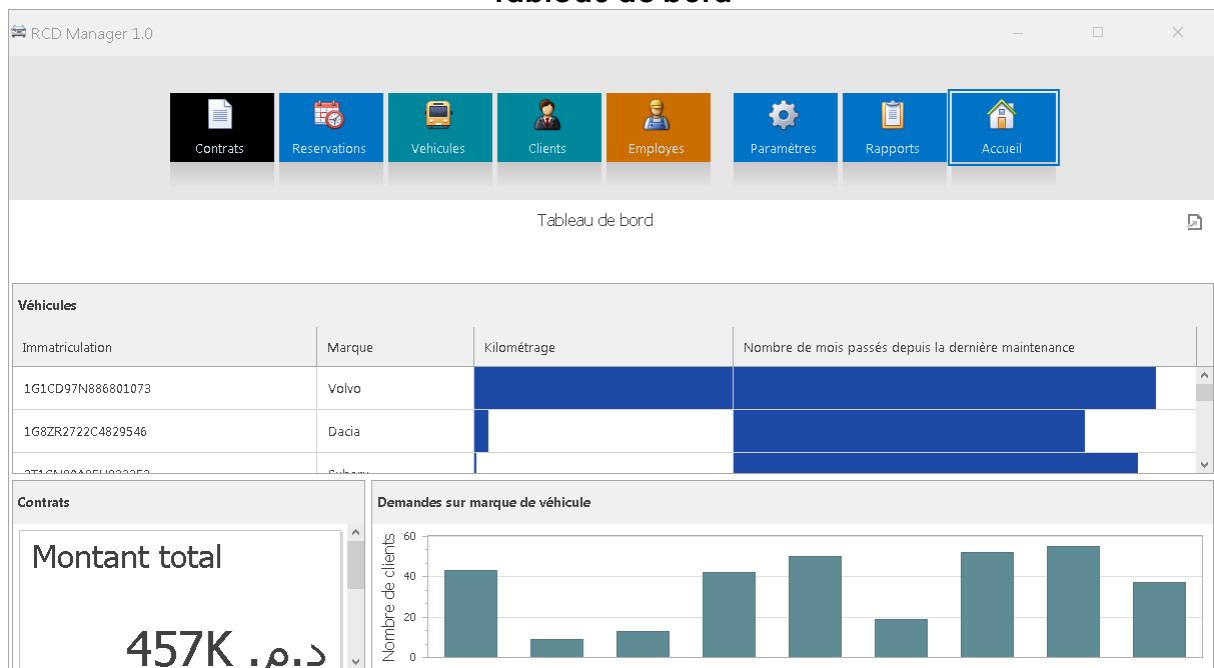
- **La couche modèle de vue** est contenue dans le dossier *ViewModels*. Ce dossier contient des sous-dossiers nommés d'après les différentes tables et vues issues de la source de données.
- **La couche vue** est contenue dans le dossier *Views*. Ce dossier contient également des sous-dossiers nommés d'après les différentes tables et vues issues de la source de données. La vue principale se trouve à la racine du dossier.
- Les **classes d'accès aux données (Data Access Interface)**
 - Représente une surcouche à ADO.NET permettant de simplifier grandement l'accès aux données en minimisant le code.
- Les **classes d'accès aux services (Application Programming Interface)**
 - Représente un ensemble de propriétés, collections, et méthodes, servant le côté métier de l'application.
- Le **rapport**
 - Représente les fichiers relatifs au rapport généré et à la visionneuse de documents.
- Le **tableau de bord (Dashboard)**
 - Représente les fichiers relatifs au tableau de bord généré et à l'assistant de conception de tableaux de bord.
- **Formulaires**
 - Chaque formulaire a deux fichiers qui lui sont associés. *Form1.cs*, ou tout autre nom donné, contient le code source écrit pour configurer le formulaire et ses contrôles, tels que les zones de liste et les zones de texte et répondre aux événements tels que les clics de bouton et les séquences de touches. Dans les projets Windows Forms simples, tout ou une grande partie du codage est effectué dans ce fichier. Le Concepteur Windows Forms écrit automatiquement dans le fichier *designer.cs* le code qui implémente toutes les actions exécutées en glissant-déplaçant des contrôles à partir de la boîte à outils.

Ci-dessus un aperçu de la structure générale du projet :



7. Maquettes d'écran

Tableau de bord



Gestion d'une collection (Ex : Les clients)

[illegible]

Edition d'un élément (Ex : Un client, un véhicule)

RCD Manager 1.0

Contrats Reservations Vehicules **Clients** Employes Paramètres Rapports Accueil

Client

Général Coordonnées Pièces d'identité

Identité Origine

Numéro

1

Nom

Kirkland

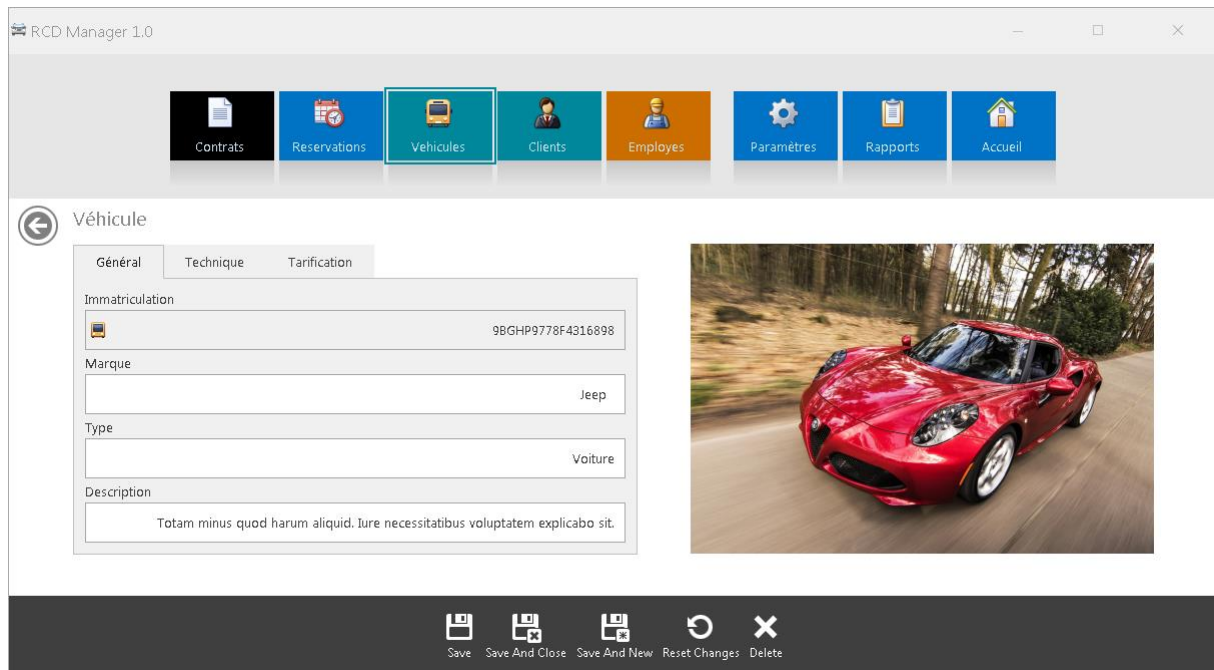
Prénom

Quentin

Type

Particulier

Save Save And Close Save And New Reset Changes Delete



Paramètres de notifications

Paramètres

Général

Vehicule

☒ Activer les notifications

Fréquence (en secondes)

10


☒ Activer l'effet sonore de notification


[Voir la liste des notifications actuelles](#)

Paramètres de véhicules

Paramètres

×

Général

Véhicule

Durée d'exploitation d'un véhicule (en années)

5

▲

▼

Première maintenance après (en mois)

12

▲

▼

Distance exigeant un vidange (en kilomètres)

10000

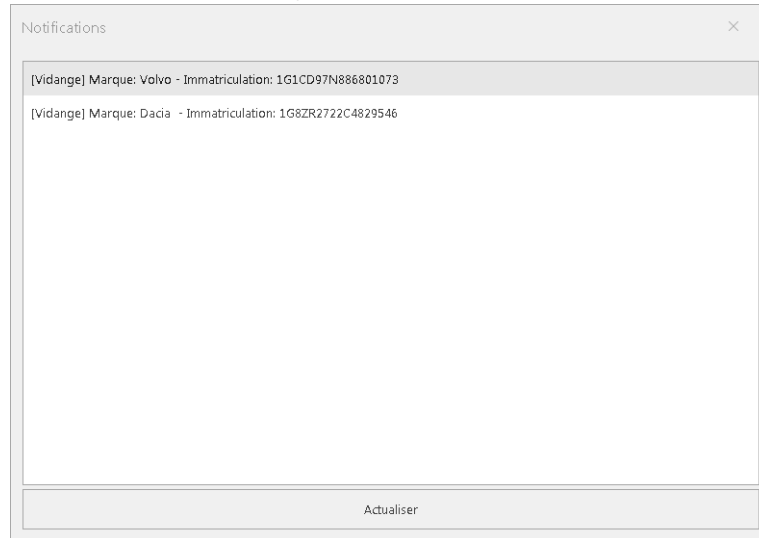
▲

▼

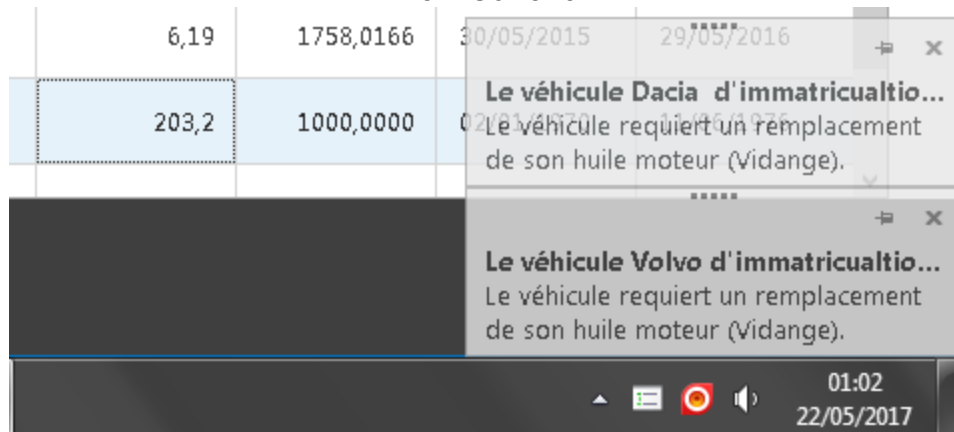
Prévisualisation et Impression (Ex : Contrat)

The screenshot shows a PDF document titled 'Prévisualisation' (Preview) in a viewer window. The document is a rental contract for RCD Service. The header features the RCD logo, which depicts a car with 'RCD' written above it and 'Rent a Car & 4 x 4' below it. To the right of the logo, the company name 'RCD Service' is displayed, followed by its address: 'Galerie Saâda, Rue Yougoslavie, N°10, Guéliz Marrakech', and contact details: 'Tél/Fax: (+212) 05 24 42 38 70'. The main title of the document is 'CONTRAT DE LOCATION'. Below the title, the contact information is repeated: 'Fetah: 06 61 75 11 58', 'Karima: 06 72 06 74 74', and 'Email: rcd.service@gmail.com'. The contract number is indicated as 'N° 1'. The section for the tenant, labeled 'LOCATAIRE', lists the following details: 'Nom: Kirkpatrick', 'Prénom: Gertha', 'Nationalité: [blank]', and 'Adresse: 87 Chapel Hill Pkwy, MidAmerican Bldg, Lansing, Michigan, 48499, West Kingston'. The PDF viewer interface includes a top menu bar with 'Fichier', 'Visualiser', and 'Arrière-plan', a toolbar with various icons, and a zoom level of 100%.

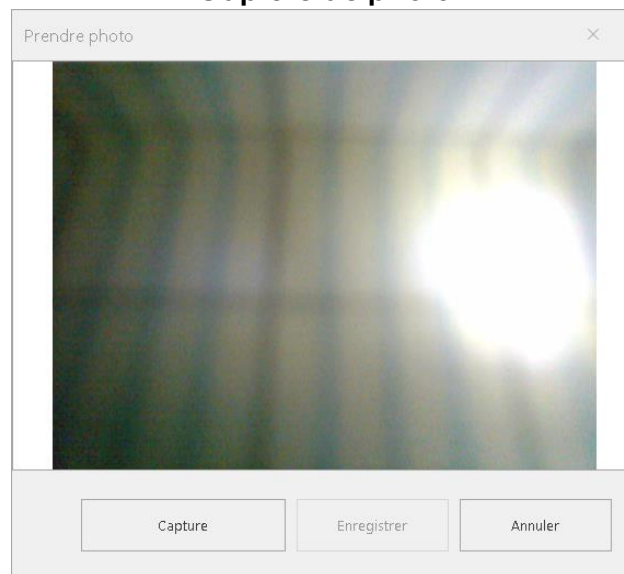
Aperçu des notifications



Notifications



Capture de photo



Exportation (Ex : Export du tableau de bord en PDF)

Exporter au format PDF - Tableau de bord

Mise en page:

☐ Portrait

☐ Paysage

☒ Auto

Taille:

Lettre

Afficher le titre:

☒

Titre:

Tableau de bord

État du filtre:

Aucun

Mode d'échelle:

Aucun

Facteur d'échelle:

1

Auto Fit Page Count:

1

Réinitialiser

Exporter

Annuler

V. Conclusion

Cette expérience professionnelle nous a permis d'approfondir et de mettre en pratique nos connaissances en terme d'analyse, de conception et de développement informatique d'applications répondant aux problématiques du marché du travail et de l'entreprise. Voici un récapitulatif de nos différents exploits durant ce projet :

- Elaboration de nos propres méthodes d'analyse inspiré de MERISE (Vision Composants et Relations).
- Expérience et connaissance plus avancée des outils acquis en formation (.NET, Visual C#, ADO.NET, etc).
- Découverte et utilisation de nouveaux outils et langages informatiques (DevExpress, Entity Framework, XML, etc).

Enfin, Nous avons constaté que l'achèvement d'un projet informatique repose en grande partie sur une organisation correcte, une planification approfondie et stratégique, et une source de motivation renouvelable.