

**DOCUMENTATION TECHNIQUE  
SOS INFOBOBO**Jonathan Borel-Jaquet. 16.05.2019. V.1.0

**Table des matières**

[1 Introduction 1](#_Toc8816565)

[2 Rappel du cahier des charges 1](#_Toc8816566)

[2.1 Objectifs 1](#_Toc8816567)

[2.2 Spécifications 1](#_Toc8816568)

[2.3 Restrictions 1](#_Toc8816569)

[2.4 Environnement 1](#_Toc8816570)

[2.5 Organisation 1](#_Toc8816571)

[2.6 Livrable 1](#_Toc8816572)

[3 Analyse fonctionnelle 1](#_Toc8816573)

[3.1 Fonctionnalités 1](#_Toc8816574)

[3.1.1 Créer une demande de réparation informatique 1](#_Toc8816575)

[3.1.2 Changer le statut d’une demande de réparation 1](#_Toc8816576)

[3.1.3 Créer un avis sur le service effectué 1](#_Toc8816577)

[3.1.4 Valider la publication d’un avis 1](#_Toc8816578)

[3.1.5 Enregistrer un rendez-vous 1](#_Toc8816579)

[3.1.6 Modifier les informations personnelles 1](#_Toc8816580)

[3.1.7 Consulter les réparations effectuées 1](#_Toc8816581)

[3.1.8 Consulter les différentes pages du site 1](#_Toc8816582)

[3.1.9 Envoyer un mail lors d’un changement de statut d’une demande 1](#_Toc8816583)

[3.1.10 Envoyer un mail lors de la création d’une nouvelle demande 1](#_Toc8816584)

[3.1.11 Se connecter à l’application 1](#_Toc8816585)

[3.2 Interfaces 1](#_Toc8816586)

[3.2.1 Barre de navigation 1](#_Toc8816587)

[3.2.2 Page d’accueil 1](#_Toc8816588)

[3.2.3 Page de contact 1](#_Toc8816589)

[3.2.4 Page d’avis 1](#_Toc8816590)

[3.2.5 Page de connexion 1](#_Toc8816591)

[3.2.6 Page calendrier 1](#_Toc8816592)

[3.2.7 Page d’administration des demandes 1](#_Toc8816593)

[3.2.8 Page de modification des informations personnelles 1](#_Toc8816594)

[3.2.9 Page d’administration des avis 1](#_Toc8816595)

[3.2.10 Page de statistique 1](#_Toc8816596)

[3.3 Cas d’utilisation 1](#_Toc8816597)

[3.3.1 Diagramme des cas d’utilisation 1](#_Toc8816598)

[4 Analyse organique 1](#_Toc8816599)

[4.1 Base de données 1](#_Toc8816600)

[4.1.1 Modèle physique de données 1](#_Toc8816601)

[4.1.2 Détails des tables 1](#_Toc8816602)

[4.2 Architecture du code 1](#_Toc8816603)

[4.2.1 Arborescence de fichier 1](#_Toc8816604)

[4.2.2 Classe POO 1](#_Toc8816605)

# Introduction

Ce document a pour but de décrire les différentes étapes à la réalisation de l’application web « Sos Infobobo », réalisé durant le TPI (Travail Pratique Individuel).

« Sos Infobobo » est une application WEB qui permet de principalement gérer des rendez-vous entre un réparateur informatique qui sera l’unique utilisateurs connecté et des potentiels clients qui seront des utilisateurs non-connecté. L’application permet également au client de poster des avis sur la potentiel réparation effectué qui devra d’abord être validé par le réparateur. Le réparateur à quant à lui accès à un calendrier avec ses différents rendez-vous planifié et des statistiques sur ses réparations effectuées une fois connecter, il aura également la possibilité de modifier ses informations personnelles dans une page dédier à cela.

# Rappel du cahier des charges

## Objectifs

Durant 11 jours de travail soit 88 heures, le but est de concevoir une application web qui permettra à un utilisateur non-connecté de faire une demande de réparation informatique au réparateur du site qui est l’unique utilisateur connecté du site. Le réparateur pourra alors lire sa demande et choisir si cette demande entre dans ses compétences ou non. Le réparateur devra valider les potentiels avis poster sur le site et aura accès à différentes statistique et modification du site web.

## Spécifications

L’application permettra de :

* Faire une demande de réparations informatique
* Visionner le statut et le détail de toute les demandes de réparation informatique
* Poster un avis sur la potentiel réparation
* Valider un avis précédemment posté par un client pour sa publication
* Modifier les informations personnelles de la page d’accueil
* Inscrire différents rendez-vous planifié dans un calendrier
* Consulter les statistiques des réparations effectuées
* Se connecter à l’application
* Informer par mail le réparateur d’une nouvelle demande
* Informer par mail le client de tout changement de statut d’une demande

## Restrictions

L’utilisateur non-connecté peut :

* Faire une demande de réparations informatique
* Poster un avis sur la potentiel réparation
* Informer par mail le réparateur lors de la création d’une nouvelle demande

L’utilisateur connecté peut :

* Se connecter à l’application
* Visionner le statut et le détail de toute les demandes de réparation informatique
* Valider un avis précédemment posté par un client pour sa publication
* Modifier les informations personnelles de la page d’accueil
* Inscrire différents rendez-vous planifiés dans un calendrier
* Consulter les statistiques des réparations effectuées
* Informer par mail le client lors d’un changement de statut d’une demande

## Environnement

* Un PC standard école avec Windows 10, 2 écrans
* Serveur Web et SGBD à choix (EasyPHP, Wamp, Laragon, Adminr, phpMyAdmin, autre)
* IDE à choix (NetBeans, Visual Studio Code, PHPStorm, autre)
* Logiciel de création de schémas (Visio, Gliffy, autre)
* Outil de versionnage de code (Git, avec dépôt distant sur Github / Bitbucket / GitLab)
* Navigateur web (Mozilla Firefox/Google Chrome)
* Logiciel de création de maquettes d’interfaces utilisateur (Sketch, Adobe XD, Proto.io, autre)
* Outil bureautique à choix pour les documents (Google Docs, MSOffice, OpenOffice)

## Organisation

Élève :

* Jonathan Borel-Jaquet, [jonathan.brljq@eduge.ch](mailto:jonathan.brljq@eduge.ch)

Maitre d’apprentissage :

* Jasmina Travnjak, [jasmina.travnjak@edu.ge.ch](mailto:jasmina.travnjak@edu.ge.ch)

Experts :

* Carol Quarroz, [cquarroz@gmail.com](mailto:cquarroz@gmail.com)
* Nicolas Terrond, [nicolas@terrond.com](mailto:nicolas@terrond.com)

## Livrable

Pour la fin du TPI, le 23 mai 2019 :

Pour les experts et la formatrice par email :

* Planning détaillé du projet
* Rapport de projet contenant le code source au format PDF
* Journal de bord
* Résumé du TPI (1 page A4)

Pour la formatrice uniquement :

* L’accès au repository distant du projet avec les droits de « clone »
* Un readme explicitant l’installation du projet en local
* Un dump de la base de données contenant la structure ainsi qu’un set de données de test

# Analyse fonctionnelle

## Fonctionnalités

### Créer une demande de réparation informatique

Cette fonctionnalité permet à un utilisateur non-connecté d’envoyer une demande de réparation au réparateur en signalant son problème informatique. L’utilisateur non-connecté devra remplir un formulaire avec les prérequis suivants :

* Nom
* Prénom
* Email
* Numéro de téléphone
* Description du problème informatique

### Changer le statut d’une demande de réparation

Cette fonctionnalité permettra au réparateur de modifier le statut d’une demande de réparation. Il pourra changer le statut d’une demande en :

* Traitée
* Refusée

### Créer un avis sur le service effectué

Cette fonctionnalité permet à un utilisateur non-connecté de créer un avis sur la potentiel réparation effectué qui serra afficher sur une page dédiée à cela.

### Valider la publication d’un avis

Cette fonctionnalité permet à un utilisateur connecté de valider ou refuser un avis précédemment créer par un utilisateur non-connecté.

### Enregistrer un rendez-vous

Cette fonctionnalité permet à un utilisateur connecté d’inscrire une demande de réparation de type ouverte dans un calendrier, il peut choisir d’y inscrire un rendez-vous pour la récupération et/ou la reddition d’un ordinateur. Une fois le rendez-vous enregistrer, le type de réparation passe à « En cours »

### Modifier les informations personnelles

Cette fonctionnalité permet à un utilisateur connecté de modifier ses informations personnelles de la page « Accueil ». Il pourra modifier les champs suivant :

* Numéro de téléphone
* Email
* Tarif
* Description

### Consulter les réparations effectuées

Cette fonctionnalité permet à un utilisateur connecté de consulter la liste des réparations qu’il a effectué, elles seront listées par mois et par années.

### Consulter les différentes pages du site

Cette fonctionnalité permet à un utilisateur non-connecté de consulter les différentes pages du site mise à sa disposition, comme les pages :

* Accueil (Information sur le réparateur)
* Contact (Formulaire de demande de réparation)
* Avis (Liste des avis de réparation)

### Envoyer un mail lors d’un changement de statut d’une demande

Cette fonctionnalité permet à un utilisateur non-connecté d’être avertis par mail de tout changement de statut de sa demande (Réf : [3.1.2 Changer le statut d’une demande de réparation](#_Changer_le_statut)).

### Envoyer un mail lors de la création d’une nouvelle demande

Cette fonctionnalité permet à un utilisateur connecté d’être avertis par mail de la création d’une nouvelle demande (Réf : [3.1.1 Créer une demande de réparation informatique](#_Créer_une_demande)).

### Se connecter à l’application

Cette fonctionnalité permet à un utilisateur qui possède un compte de pouvoir se connecter et avoir accès aux différentes fonctionnalités d’administration.

## Interfaces

### Barre de navigation

#### Utilisateur non-connecté



4

3

2

1D

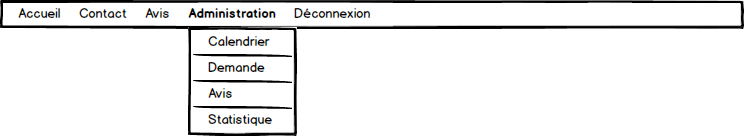
Fig. 1 - Barre de navigation d’un utilisateur non-connecté

1. Le lien « Accueil » permet à l’utilisateur d’aller à la page d’accueil (voir Fig. 3 – Page d’accueil)
2. Le lien « Contact » permet à l’utilisateur d’aller sur la page de contact (voir Fig. 4 - Page de contact").
3. Le lien « Avis » permet à l’utilisateur d’aller sur la page d’avis (voir Fig. 5 - Page d’Avis).
4. Le lien « Connexion » permet à l’utilisateur d’aller sur la page de connexion du site (voir Fig. 6 - Page de connexion).

#### Utilisateur connecté

6D

1D



4D

5D

3

2D

Fig. 2 - Barre de navigation d'un utilisateur connecté

1. Le lien « Administration » permet à l’utilisateur d’ouvrir le menu déroulant d’administration (voir Fig. 2 - Barre de navigation d'un utilisateur connecté).
2. Le lien « Calendrier » permet à l’utilisateur d’accéder à la page de planification de rendez-vous (voir Fig. 7 - Page calendrier)
3. Le lien « Demande » permet à l’utilisateur d’accéder à la page répertoriant toute les demandes (voir Fig. 8 - Page d’administration des demandes).
4. Le lien « Avis » permet à l’utilisateur d’accéder à la page listant les avis en attente de validation (voir Fig. 10 - Page d'administration des avis).
5. Le lien « Statistique » permet à l’utilisateur d’accéder la page de statistique (voir Fig. 11 - Page de statistique).
6. Le lien « Déconnexion » permet à l’utilisateur de se déconnecter du site.

### Page d’accueil

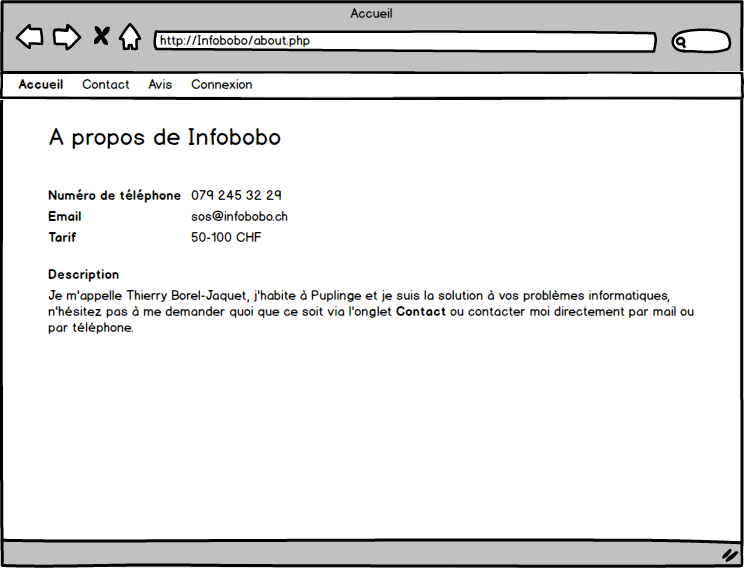
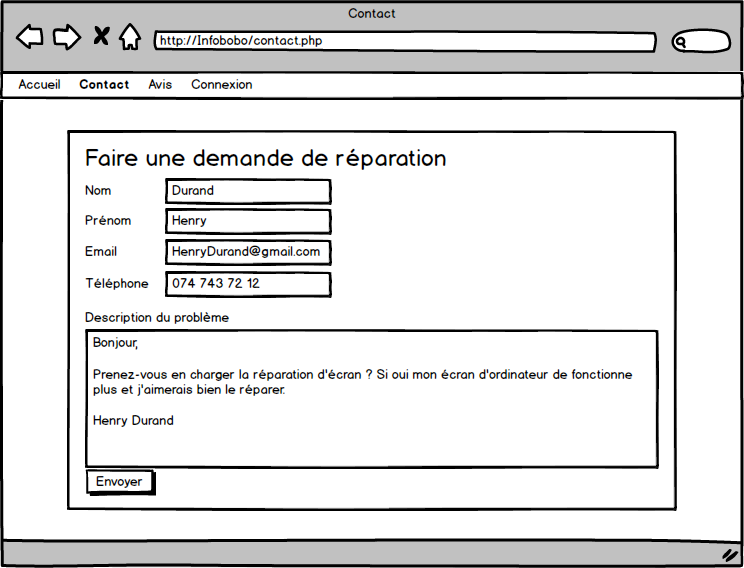


Fig. 3 - Page d’accueil

La page d’accueil accueille l’utilisateur et lui permet de consulter les informations personnelles du réparateur.

### Page de contact



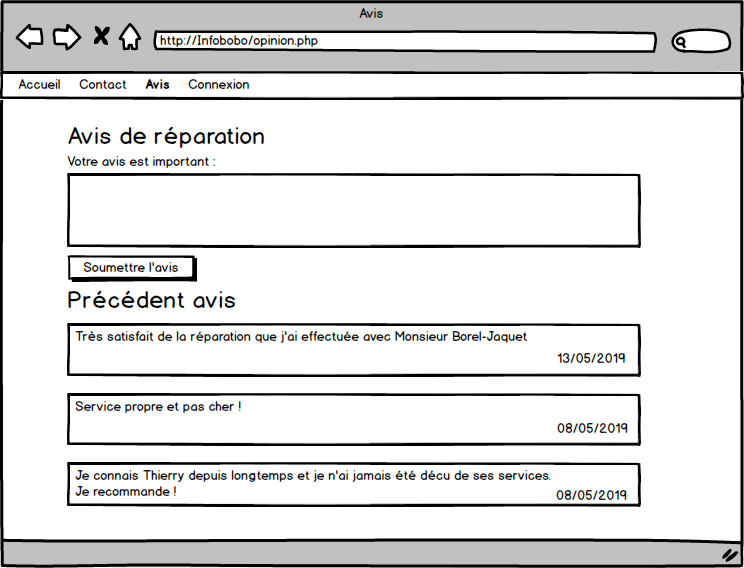
1D

Fig. 4 - Page de contact

La page « Contact » permet à l’utilisateur de créer une demande qui sera soumis au réparateur.

1. Le bouton « Envoyer » permet d’envoyer la demande au réparateur, un pop-up confirme si l’envoie de la demande à bien été effectué ou non.

### Page d’avis



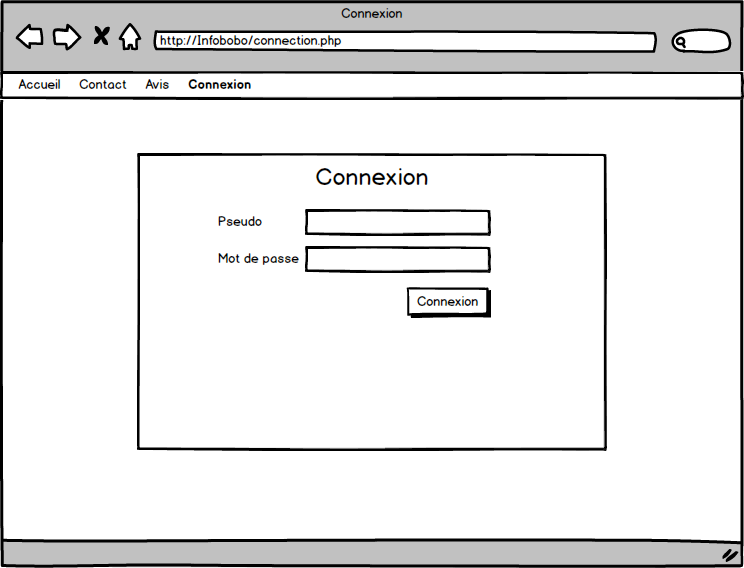
1D

Fig. 5 - Page d’avis

La page « Avis » permet à l’utilisateur d’avoir la possibilité de soumettre un avis sur la réparation effectuée, il peut également consulter les précédents avis postés sur le site.

1. Le bouton « Soumettre l’avis » permet d’envoyer l’avis à l’administrateur qui devra ensuite le valider afin d’être affiché, un pop- confirmera si l’envoie de l’avis à bien été effectué ou non.

### Page de connexion



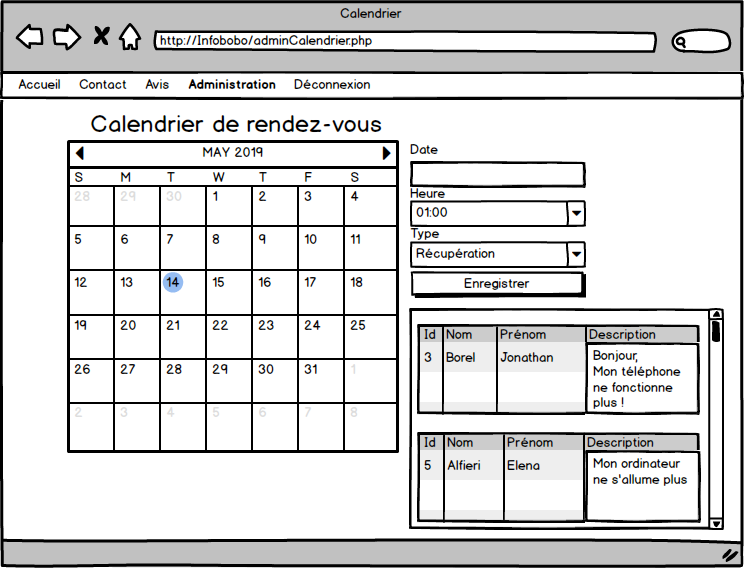
1D

Fig. 6 - Page de connexion

La page de connexion permet à un utilisateur de se connecter à l’application.

1. Le bouton « Connexion » permet de connecter l’utilisateur à l’application, un pop-up confirmera si la connexion à réussi ou si le pseudo ou le mot de passe est incorrect.

### Page calendrier



2

3D

5D

4D

6D

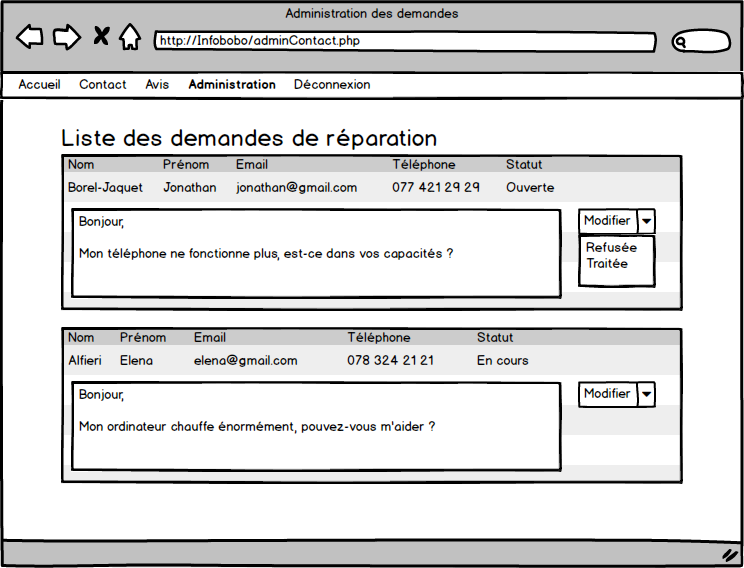
1D

Fig. 7 - Page calendrier

La page « Calendrier » permet au réparateur de planifier ses rendez-vous de récupération ou de réédition.

1. Une fois une date sélectionnée, le champ date (réf .2) est remplis avec la date choisit
2. Zone de texte non modifiable contenant la date choisis par le réparateur.
3. Liste de choix pour l’heure du rendez-vous à planifier.
4. Liste de choix pour le type de rendez-vous (Récupération ou reddition)
5. Le bouton « Enregistrer » permet au réparateur d’enregistrer un rendez-vous une fois les informations bien sélectionné (réf .2 réf .6)
6. Liste de tableau correspondant aux demandes de type ouverte, le réparateur doit cliquer sur l’une d’entre elle pour la sélectionner.

### Page d’administration des demandes



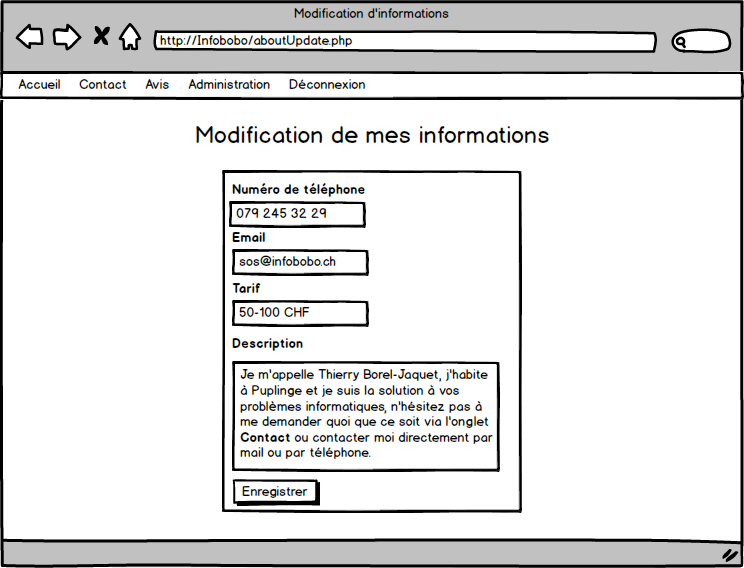
1D

Fig. 8 - Page d’administration des demandes

La page d’administration des demandes permet au réparateur de consulter toute ses demandes et de modifier leur statut.

1. Le bouton « Modifier » permet au réparateur de choisir un nouveau statut pour la demande (Traitée ou Refusée)

### Page de modification des informations personnelles



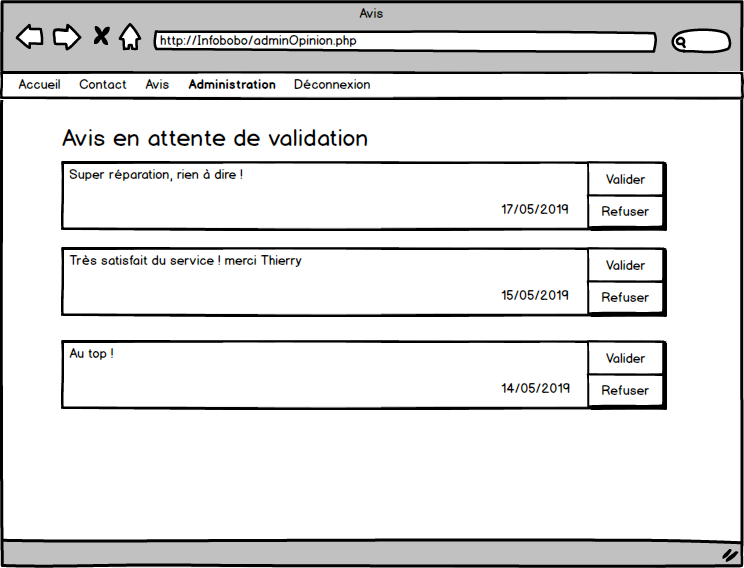
1D

Fig. 9 - Page de modification des informations personnelles

La page de modification des informations personnelles permet au réparateur de modifier les informations de la page accueil du site (voir Fig. 3 - Page d’accueil).

1. Le bouton « Enregistrer » permet au réparateur de modifier ses informations personnelles pour la page d’accueil du site.

### Page d’administration des avis



2D

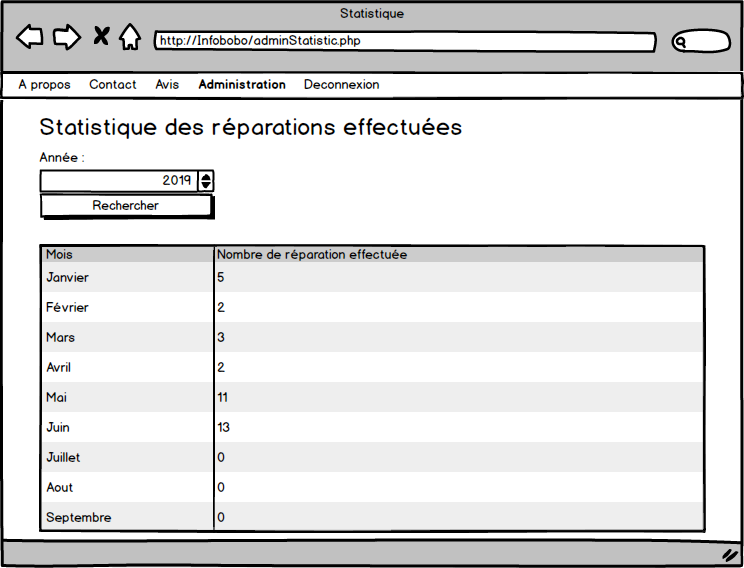
1D

Fig. 10 - Page d'administration des avis

La page d’administration des avis permet au réparateur de valider ou non un avis en attente de publication.

1. Le bouton « Valider » permet au réparateur de valider l’avis
2. Le bouton « Refuser » permet au réparateur de refuser l’avis

### Page de statistique



1D

2D

Fig. 11 - Page de statistique

La page statistique permet au réparateur de consulté le nombre de réparation faite par mois durant une année sélectionnée.

1. Le numérique « Up Down » permet au réparateur de sélectionné l’année à laquelle il veut consulter le nombre de réparation
2. Le bouton « Rechercher » permet au réparateur d’afficher le nombre de réparation par mois de l’année précédemment sélectionné

## Cas d’utilisation

### Diagramme des cas d’utilisation

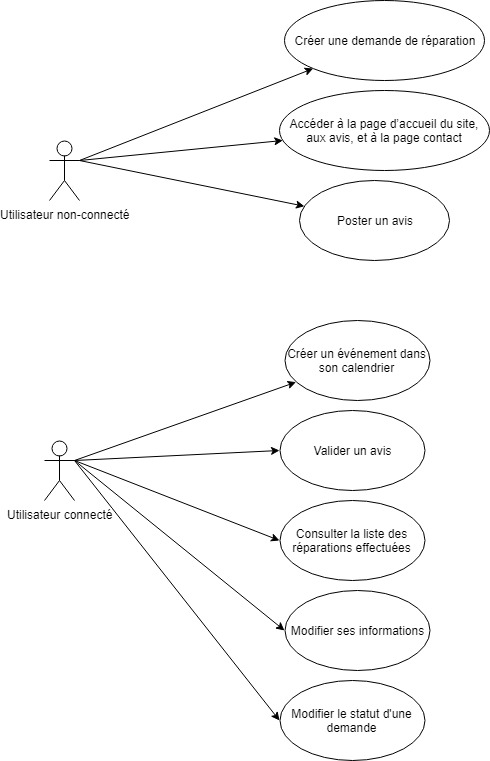


Fig. 12 - Diagramme des cas d'utilisation

# Analyse organique

## Base de données

### Modèle physique de données

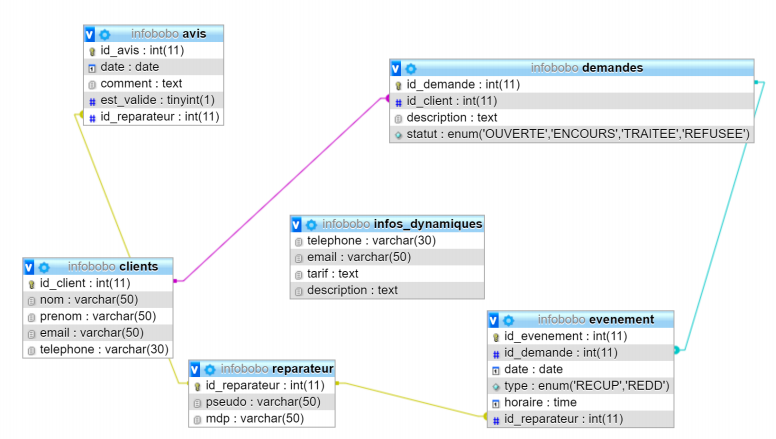


Fig. 13 - Modèle physique de données initial

### Détails des tables

|  |  |
| --- | --- |
| **Abréviation utilisée** | **Signification** |
| PK | Clé primaire |
| NN | Non Nul (Empêche la valeur NULL) |
| UQ | Unique (Empêche que la même valeur soit deux fois dans la table) |
| AI | Auto incrément |

#### Table « clients »

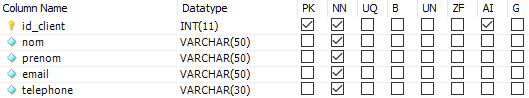


Fig. 14 - Table « clients »

#### Table « avis »

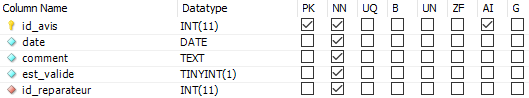


Fig. 15 - Table « avis »

#### Table « demandes »



Fig. 16 - Table « demandes »

#### Table « reparateur »

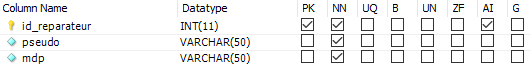


Fig. 17 - Table « reparateur »

#### Table « evenement »

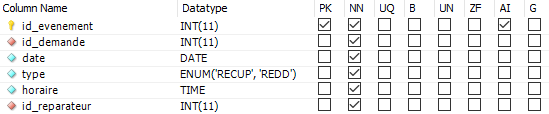


Fig. 18 - « Table evenement »

#### Table « infos\_dynamiques »

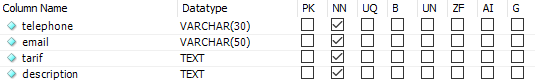


Fig. 19 - Table « infos\_dynamiques »

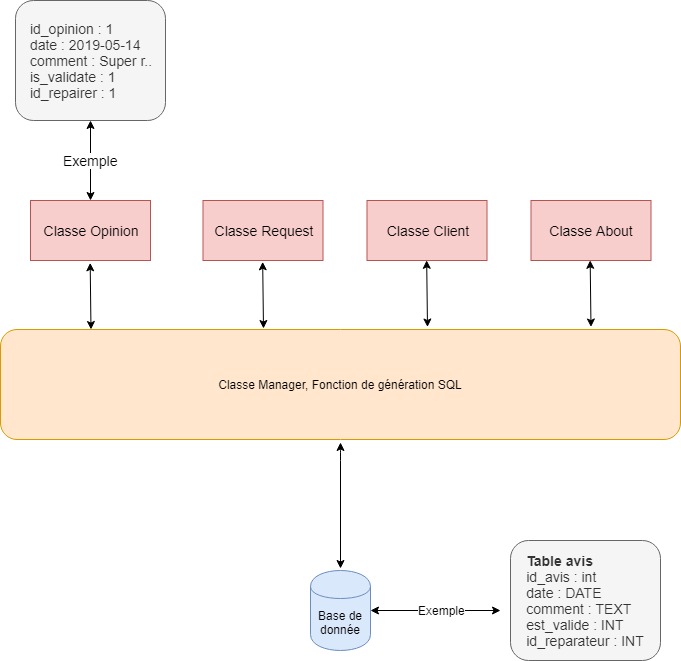
## Architecture du code

### Arborescence de fichier

### Classe POO

Dans le cadre de mon développement, j’ai décidé de créer des classes avec les tables dont j’avais besoin de ressortir des informations, c’est une sorte de conteneur. Cela permet à mon code de ne pas être trop dépendant de la base de donnée lors de la récupération de mes informations SQL.

J’ai également utiliser des manager (classe statique) qui me permettent de structurer mes fonctions et de le rendre mieux réutilisable. Une fonction permettant la connexion de l’utilisateur se trouvera dans le manager « UserManager » par exemple.

   
  
Fig. 20 – Digramme du fonctionnement de l’interaction avec la base de données

### Classes utilisées

#### Classe About

Cette classe permet de stocker les informations de la table « infos\_dynamiques ».

|  |  |  |
| --- | --- | --- |
| **Class About** | | |
| **Variable** | **Type** | **Description** |
| id\_about | int | L’identifiant des informations dynamiques |
| phoneNumber | string | Le numéro de téléphone du réparateur |
| email | string | L'email du réparateur |
| price | string | Le prix du potentiel service |
| description | string | La description du réparateur et de ses services |

Exemple d’utilisation 1

**$a** = **new** About();

**$a**->**id\_about** = 1

**$a**->**phoneNumber** = "077 421 34 20";

**$a**->**email** = "sos@infobobo.ch";

**$a**->**price** = "50 - 100 CHF";

**$a**->**description** = "Bonjour, je m'appelle Thierry Borel-Jaquet et je suis la solut..";

Exemple d’utilisation 2

**$a** = **new** About(1,"077 421 34 20","sos@infobobo.ch","50 - 100 CHF","Bonjour, je m'appelle Thierry Borel-Jaquet et je suis la solut..");

#### Classe Client

Cette classe permet de stocker les informations de la table « clients ».

|  |  |  |
| --- | --- | --- |
| **Class Client** | | |
| **Variable** | **Type** | **Description** |
| id\_client | int | L'identifiant du client |
| firstName | string | Le nom du client |
| secondName | string | Le prénom du client |
| email | string | L'email du client |
| phoneNumber | string | Le numéro de téléphone du client |

Exemple d’utilisation 1

**$c** = **new** Client();

**$c**->**id\_client** = 2;

**$c**->**firstName** = "Coptère";

**$c**->**secondName** = "Ellie";

**$c**->**email** = "elli.coptere@gmail.com";

**$c**->**phoneNumber** = "077 421 34 20";

Exemple d’utilisation 2

**$c** = **new** Client(2,"Coptère","Ellie","elli.coptere@gmail.com","077 421 34 20");

#### Classe Opinion

Cette classe permet de stocker les informations de la table « avis ».

|  |  |  |
| --- | --- | --- |
| **Class Opinion** | | |
| **Variable** | **Type** | **Description** |
| id\_opinion | int | L'identifiant de l’avis |
| date | string | La date de création de l’avis |
| comment | string | Le commentaire de l’avis |
| is\_validate | bool | Si l’avis est validé ou non |
| id\_repairer | int | L’identifiant du réparateur concerné |

Exemple d’utilisation 1

**$o** = **new** Opinion();

**$o**->**id\_opinion** = 2;

**$o**->**date** = "27/03/19";

**$o**->**comment** = "Super réparation de Monsieur Borel-Jaquet, au top !";

**$o**->**is\_validate** = FALSE;

**$o**->**id\_repairer** = 1;

Exemple d’utilisation 2

**$o** = **new** Opinion(2, "27/03/19", "Super réparation de Monsieur Borel-Jaquet, au top !",FALSE,1);

#### Classe Request

Cette classe permet de stocker les informations de la table « demandes ».

|  |  |  |
| --- | --- | --- |
| **Class Request** | | |
| **Variable** | **Type** | **Description** |
| id\_request | int | L’identifiant de la demande |
| id\_client | int | L’identifiant du client ayant fait la demande |
| description | string | Description du problème |
| status | string | Statut de la demande |

Exemple d’utilisation 1

**$r** = **new** Request();

**$r**->**id\_request** = 2;

**$r**->**id\_client** = 4;

**$r**->**description** = "Bonjour, mon ordinateur à fréquemment des écrans bleus, est-ce dans vos compétences ?";

**$r**->**status** = "OUVERTE";

Exemple d’utilisation 2

**$r** = **new** Request(1, 4, "Bonjour, mon ordinateur à...","OUVERTE");

#### AboutManager

La classe « AboutManager » contient les fonctions en rapport avec les informations personnelles du réparateur, plus précisément de la table « infos\_dynamiques »

|  |  |
| --- | --- |
| **Nom de la fonction** | **Description** |
| GetAboutInformation() | Retourne un objet About contenant les informations personnelles du réparateur. |
| UpdateAboutInformation(**$phoneNumber**, **$email**, **$price**, **$description**) | Modifie les informations personnelles du réparateur avec de nouvelles valeur passé en paramètre. |

#### ClientManager

La classe « ClientManager » contient les fonctions en rapport avec les clients et leurs informations, plus précisément de la table « clients ».

|  |  |
| --- | --- |
| **Nom de la fonction** | **Description** |
| AddClient(**$firstName**, **$secondName**, **$email**, **$phoneNumber**) | Enregistre les informations d’un client dans la base de données |

#### EventManager

La classe EventManager contient les fonctions en rapport avec les événements créés par le réparateur, plus précisément de la table « evenement ».

|  |  |
| --- | --- |
| **Nom de la fonction** | **Description** |
| AddEvent(**$id\_request**, **$dateEvent**, **$typeEvent**, **$hourEvent**) | Enregistre un événement et modifie le statut d'une demande dans la base de données. |
| GetAllEventFormatJSON() | Récupère tous les événements en format JSON destiné au calendrier JavaScript FullCalendar |

#### MailerManager

La classe MailerManager contient les fonctions permettant d’envoyer un mail en utilisant la librairie Swift Mailer.

|  |  |
| --- | --- |
| **Nom de la fonction** | **Description** |
| SendMail**($desinataire**, **$subject**, **$textMessage**) | Enregistre un événement et modifie le statut d'une demande dans la base de données. |

#### OpinionManager

La classe OpinionManager contient les fonctions en rapport avec les avis créés par les clients, plus précisément de la table « avis »

|  |  |
| --- | --- |
| **Nom de la fonction** | **Description** |
| SendMail($desinataire, $subject, $textMessage) | Enregistre un événement et modifie le statut d'une demande dans la base de données. |