2022

NSI Prospect

App Android

OLLIVIER CHRIS – Lopes Jorys – Chabrier Mathis

# Page de service

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Auteur** | **Date** | **Description** |
| 1.0 | OLLIVIER Chris | 18/01/2022 | Création mise en page + |

Table des matiÈres

[Page de service 1](#_Toc102341251)

[Objet du document 3](#_Toc102341252)

[Contexte 3](#_Toc102341253)

[Gestion des utilisateurs 3](#_Toc102341254)

[Infrastructure Réseau 4](#_Toc102341255)

[Backend 5](#_Toc102341256)

[API PHP 5](#_Toc102341257)

[Base de donnée 7](#_Toc102341258)

[Maquette 8](#_Toc102341259)

[CONNEXION utilisateur 8](#_Toc102341260)

[MENU principal 9](#_Toc102341261)

[Consultation, modification et saisie d'un prospect 10](#_Toc102341262)

# Objet du document

Ce document contient des éléments d'analyse, de conception et de réalisation d'un outil de gestion des visiteurs. L'objectif est de conserver la trace et documenter le premier contact avec un prospect potentiel.

# Contexte

Fondée en 1988, NSI (Networking Solutions Incorporated) est une ESN (Entreprise de Services du Numérique). NSI possède les compétences et l'expérience pour intervenir sur la totalité du cycle de vie des solutions applicatives de ses clients : de l'analyse-conception jusqu'à l'exploitation en passant par la réalisation, le déploiement et la maintenance.

NSI est organisé autour de trois pôles de compétence :

1. Le développement d'applications pour ses clients.
2. L'édition logicielle en particulier pour les mobiles sous Android.
3. La conception et implémentation d'infrastructures informatiques.

# Gestion des utilisateurs

Mark TWAIN, le Directeur commercial de NSI, fait un constat très simple :

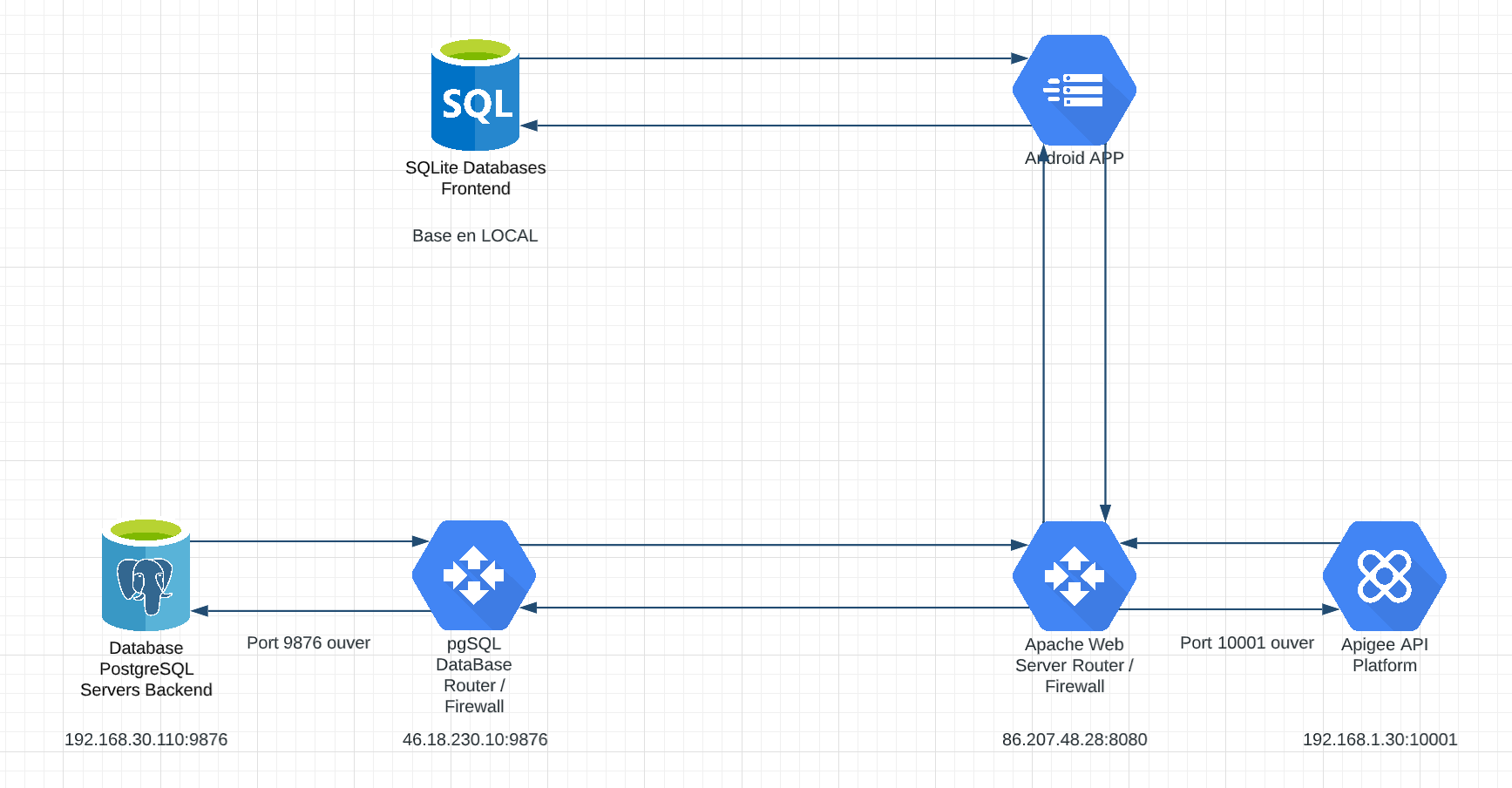
NSI expose son savoir-faire lors de salons professionnels. Ces salons, foires, expositions, congrès, forums, conférences sont l'occasion de rencontres informelles avec des représentants d'entreprises qui deviendront peut-être des prospects voire des clients.

Ces contacts sont purement informels et sous l'entière responsabilité du collaborateur qui réalise physiquement ce contact sans que jamais NSI n'en soit informé. Concrètement et dans le meilleur des cas, l'ingénieur commercial prend des notes personnelles dans un cahier ou une application locale type bureautique. Aucune trace n'est conservée au niveau de l'entreprise. Ce n'est que plus tard lorsque le contact deviendra un prospect ou un client que NSI commencera à prendre en charge la gestion des informations utiles au travers d'une solution de CRM (Customer Relationship Management – gestion de la relation client)

Ce qui vaut pour NSI vaut également pour toutes les entreprises et organisations qui souhaitent gérer le premier contact avec une personne. Il n'existe aucune application sur le marché prenant en charge ce besoin. Ce que l'on trouve sur le marché en matière de gestion des visiteurs se résume à la gestion de la sécurité ou de la billettique !

Fort de ce constat NSI développera une solution mobile en Java sous Android pour prendre en charge le premier contact avec une personne : par exemple un visiteur sur un salon, une exposition, un musée, un stade, etc…

# Infrastructure Réseau



L’infrastructure pour l’application et plutôt simple avec un backend avec une API en PHP et Base de donnee postgreSQL et un frontend avec android pour l’application et une base de donnee SQLite en local.

# Backend

Le back permet à l’application mobile de récupérer les données de la base de données distantes. Elle passe donc par une appli qui est développé en PHP. Qui va récupérer les données dans une base de données, poserait SQL.

## API PHP

Se diagramme de classes présente la structure des classes de notre API. Elle se compose donc des trois classes dont Dbh qui permet de crée la connexion vers la base de données cette class est hérite par les classes User et Prospect qui sont deux classes qui permet des faire du CRUD dans la base de donnes.

class Dbh {  
 protected function connect() {  
 try {  
 return $dbh = new PDO("pgsql:host=46.18.230.10;port=9876;dbname=NSI\_prospect-Ollivier\_chris;"

,"postgres", "P@ssw0rdsio");  
 }catch (PDOException $exception){  
 print 'Error :' . $exception->getMessage() .' <br>';  
 die();  
 }  
 }  
}

La class Dbh n’a qu’une seul méthode qui permet de se connexion a la base de données, elle est héritée par toutes les classes qui ont le besoin de se connecte.

La class Prospect a donc 5 méthodes. Elles sont les suivent :

public function getOneProspect($raisonSocial){  
  
 $raisonSocial = strtolower($raisonSocial);  
  
 $stmt = $this->connect()->prepare('select \* from "Prospect" where "raisonSocial" = ?;');  
  
 if (!$stmt->execute(array($raisonSocial))) {  
 $stmt = null;  
 header("location: ./?error=404");  
 exit();  
 }  
 return $stmt->fetchAll(PDO::*FETCH\_ASSOC*);  
}

Cette méthode permet de récupère un prospect avec sa raison social. Elle lui retourne un array avec toutes les info.

public function getAllProspect(){  
 $stmt = $this->connect()->prepare('select \* from "Prospect";');  
  
 if (!$stmt->execute()) {  
 $stmt = null;  
 header("location: ./?error=404");  
 exit();  
 }  
 return $stmt->fetchAll(PDO::*FETCH\_ASSOC*);  
}

Cette méthode permet de récupère toutes les prospects. Elle lui retourne un array avec toutes les prospect et leurs info.

public function addOneProspect($id,$nom,$prenom,$siret,$score,$raisonSocial,$mail,$tel){  
 $stmt = $this->connect()->prepare('insert into "Prospect"(id,nom,prenom,siret,score,raisonSocial,mail,tel) values (?,?,?,?,?,?,?,?)');  
  
 if (!$stmt->execute(array($id,$nom,$prenom,$siret,$score,$raisonSocial,$mail,$tel))) {  
 $stmt = null;  
 header("location: ./?error=404");  
 exit();  
 }  
 $stmt = null;  
}

Cette méthode permet d’ajouter un prospect avec toutes les info nécessaire. Elle ne retourne rien.

public function updateOneProspect($id,$nom,$prenom,$siret,$score,$raisonSocial,$mail,$tel){  
 $stmt = $this->connect()->prepare('update "Prospect" set nom = ?,prenom = ?,siret = ?,score = ?,raisonSocial = ?,mail = ?,tel = ? where id = ?');  
  
 if (!$stmt->execute(array($nom,$prenom,$siret,$score,$raisonSocial,$mail,$tel,$id))) {  
 $stmt = null;  
 header("location: ./?error=404");  
 exit();  
 }  
 $stmt = null;  
}

Cette méthode permet de modifier un prospect avec toutes les info nécessaire.

public function checkIfExist($id){  
 $stmt = $this->connect()->prepare('select \* from "Prospect" WHERE id = ? ');  
  
 if (!$stmt->execute(array($id))){  
 $stmt = null;  
 header("location: ../frontend/php/index.php#popup-one?error=stmtFailed");  
 exit();  
 }  
  
 if ($stmt->rowCount() > 0){  
 $resCheck =false;  
 }else{  
 $resCheck =true;  
 }  
  
 return $resCheck;  
}

Cette méthode permet de savoir si un utilisateur existe si oui elle retourne un INT supérieur à un. La class User permet la même chose que la class Prospect mais avec les données.

## Base de donnée

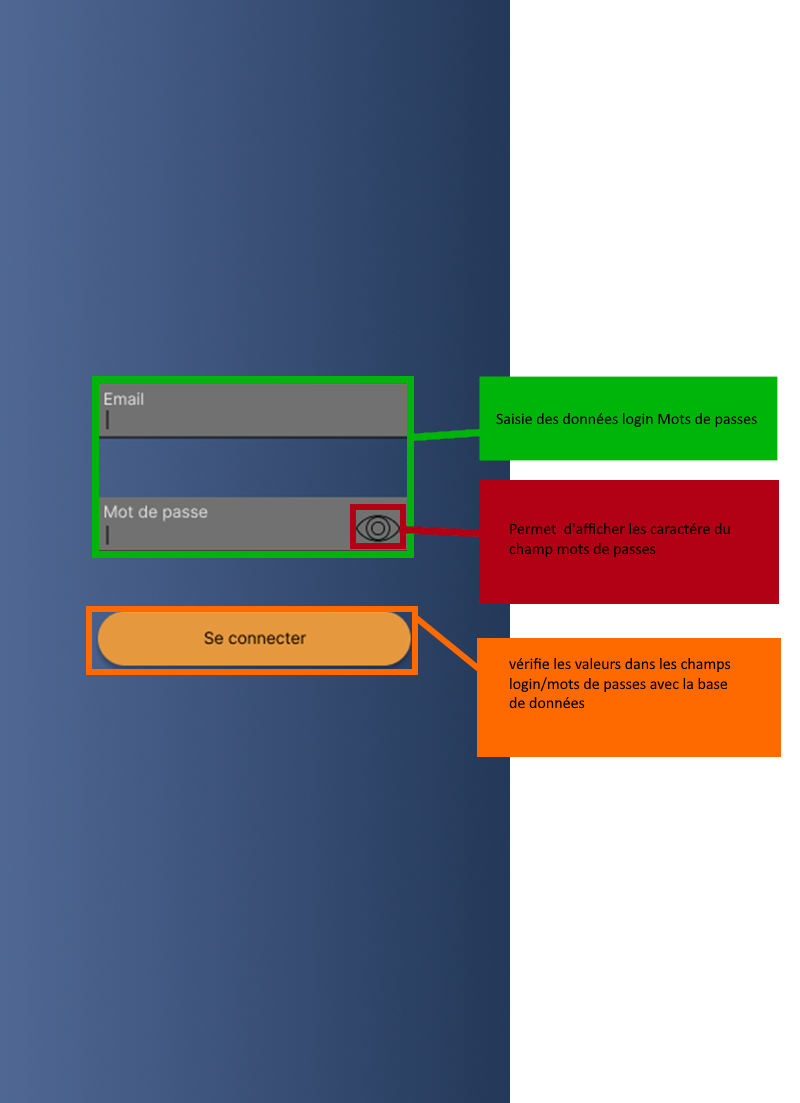
Une image contenant texte, noir, capture d’écran, tableau de points

Description générée automatiquementla base de données est très simple elle est la même pour le frontend car ont veut que les données soit les même de partout.

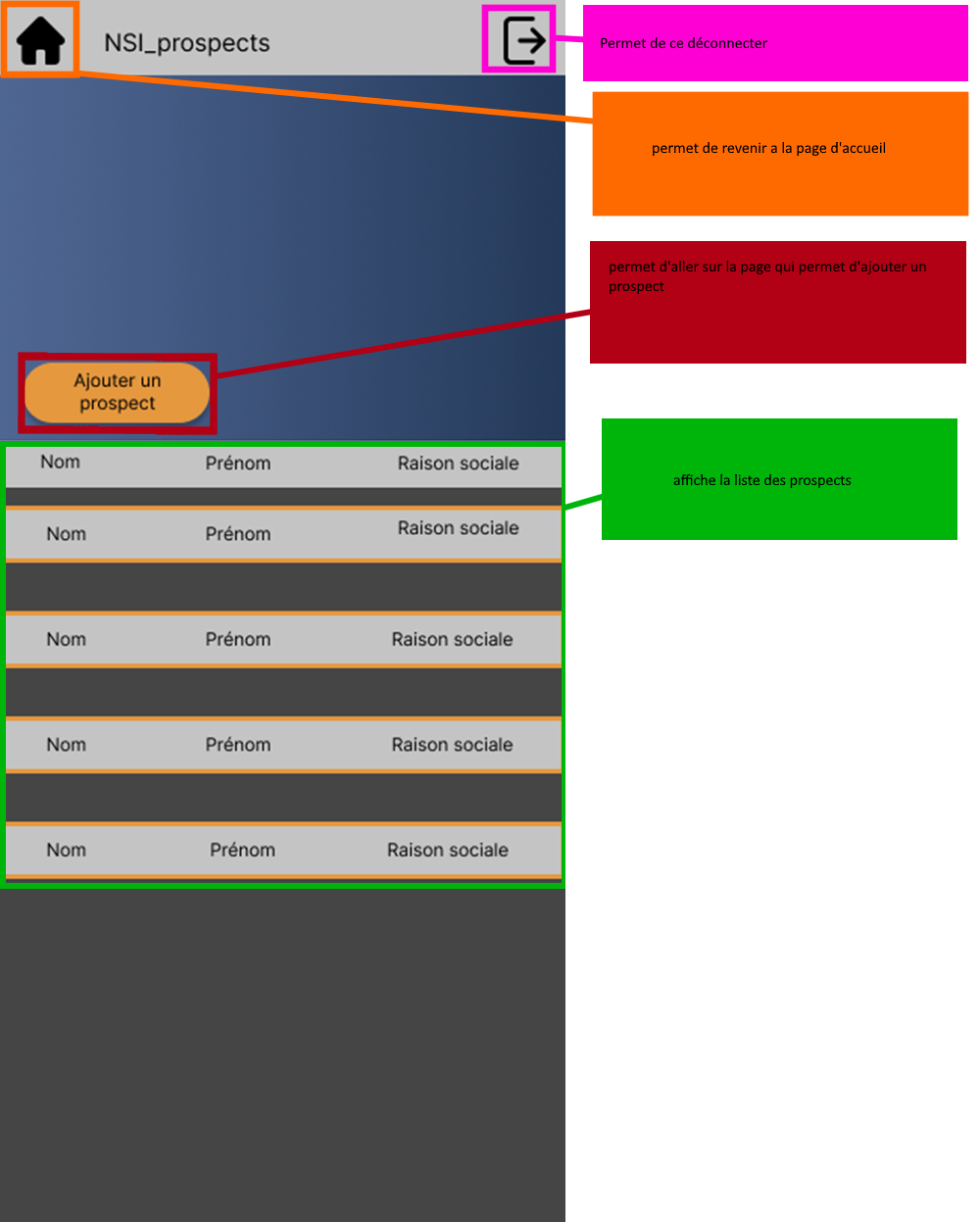
# Maquette

Cette maquette représente l’interface utilisateur de l’appication.

## CONNEXION utilisateur



## MENU principal



## Consultation, modification et saisie d'un prospect

