Projet Web Polytech 2017

CREATION DU SITE DE KG-ENTREPRISE

AGBODJOGBE YVES-ALAIN

REMERCIEMENT

Je tenais à remercier mes professeurs, madame Anne Laurent et monsieur Arnaud Castelltort pour l'expérience qu'ils m'ont apporté à travers la réalisation de ce projet. Egalement mes collègues futurs ingénieurs en IG3 pour les différents types d'aides qu'ils ont pu m'apporter pendant toute la durée de ce projet.

Table des matières

INTRODUCTION

Dans les cadre de mon projet de troisième année portant sur la création d'une application web, j'ai choisi de me pencher sur une problématique que m'ont posés des amis voulant monter une start-up basée sur la création de business plan et le consulting (analyse comptable, assistance fiscale). Ces derniers voulaient avoir un site de présentation, je leur ai donc proposé d'intégrer une gestion des clients et des contrats sur ce site, ce qui leur à plu. Le projet s'est alors mis en marche.

Dans ce rapport nous allons éplucher cette application web, voir les technologies utilisées, pourquoi ces technologies ont été utilisées plutôt que d'autres, l'implantation du site sur le Cloud et beaucoup d'autres aspects dont mon ressentit à la fin de ce projet (Post mortem). Allons-y dans la découverte du site web **KG-Entreprisews.**

Comment ?, Pour qui ?

1- User case

Cette application devait être aussi claire et belle que possible. En effet en plus de faire office de vitrine à une future start-up de finance, elle devait être agréable aux visiteurs et aux clients de cette start-up.

Elle devait entre autre servir à :

Pour les administrateurs (ce sont les possesseurs la start-up et certains de ceux qui y travailleront) :

- Pouvoir modifier les références d'un client (nom, description, représentant,……)
- Pouvoir supprimer ou rajouter des clients,
- Pouvoir supprimer des contrats,
- Pouvoir accéder aux informations des administrateurs (sauf le mot de passe sinon ce ne serais pas très sécurisé tout ça…..),
- Modifier les informations des administrateurs,
- Supprimer ou rajouter des administrateurs,
- Permettre un « upload » des scans des contrats lorsqu'on crée un contrat,
- Permettre de télécharger le scan d'un contrat qui est présent en base de données.

Pour les clients :

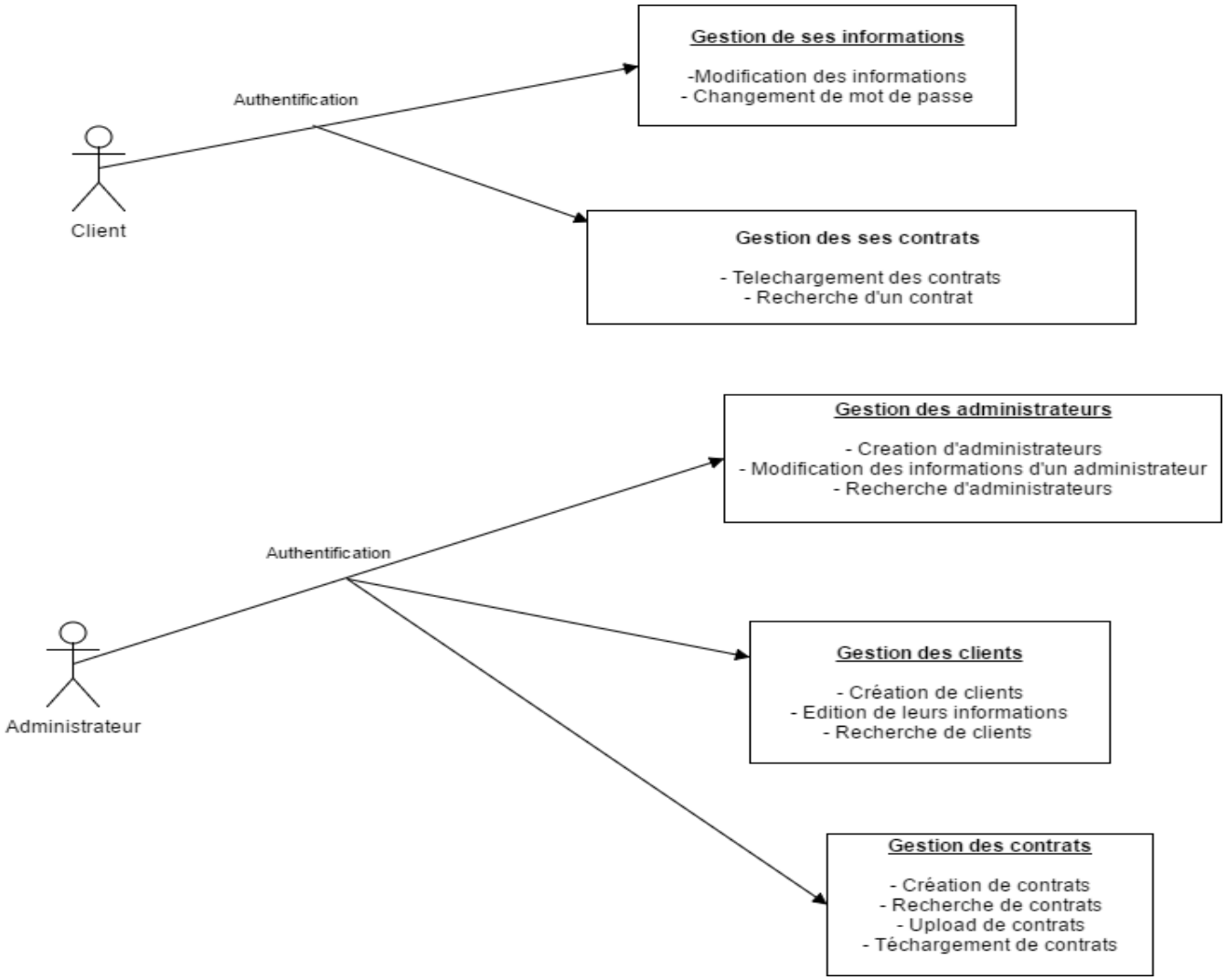
- Se connecter pour avoir accès à leurs informations,
- Pouvoir modifier ces dites informations, également leur mot de passe,
- Avoir accès au téléchargement de leurs contrats via le site.

2– Pour qui ?

J'ai déjà abordé le sujet dans la précédente partie. Comme je l'ai écrit ce site serait destiné pour la page d'accueil aux visiteurs et potentiels clients et tout ce qui se passe après la connexion, aux administrateurs du site et aux clients de l'entreprise.

Nous avons vu à qui cette application était destinée et les besoins fonctionnels qu'elle est censée remplir. Voyons maintenant à travers les parties qui vont suivre comment j'ai essayé de répondre au mieux à ce besoin.

3 – Actions permises sur le site

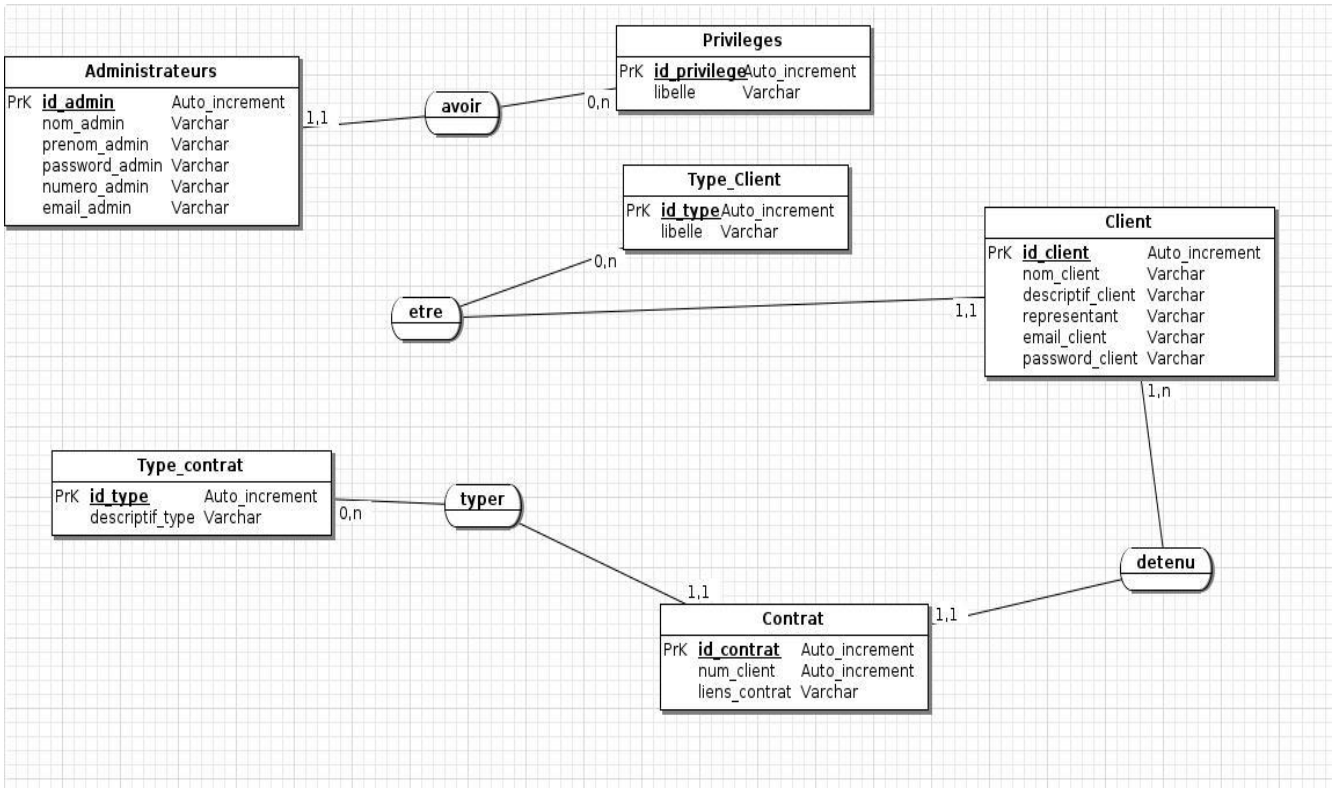


ARCHITECTURE TECHNIQUE

1- Base de données

1.1 – Modélisation

J'ai utilisé la méthode du MCD pour avoir une vision claire de ce que je voulais implanter dans ma base de données. Vous trouverez ce MCD ci-dessous.



1.2 – Liste des tables et des contraintes

Comme vous pouvez les voir sur l'image SQL dans la sous partie précédente, je suis parti sur un système de 6 tables.

Les tables principales étant celles des « Administrateurs », des « Clients » et des « Contrats ». Les tables secondaires sont les tables « Type_contrat », »Type_client » et « Privilèges ».

Pourquoi primaires et secondaires ? Je considère que les tables primaires sont les tables les plus importantes. C'est sur elles que va reposer la réponse à mes besoins fonctionnels. Les tables secondaires quant à elles sont là pour apporter des informations complémentaires à celles que je peux obtenir dans les primaires. Par exemple grâce à la table secondaire « Type_client », je sais si un client est un Particulier ou une Entreprise.

Quel est le rôle des différentes tables ?

La table « Administrateurs » contient toutes les informations qui me sont utiles pour reconnaitre un Administrateur.

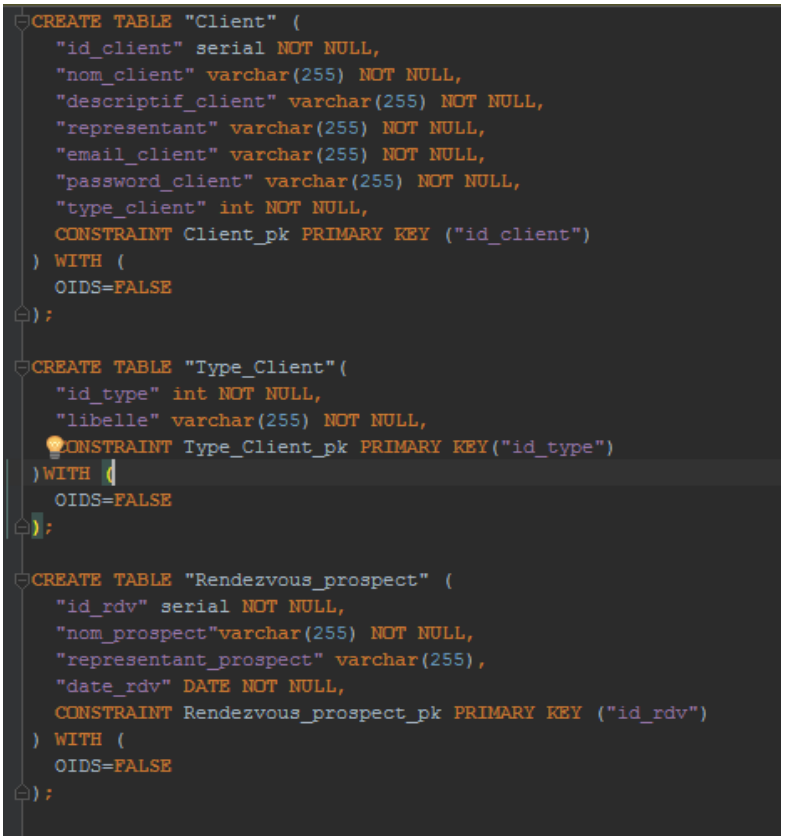
La table « Client » me permet d'avoir les informations sur les clients.

La table « Contrat » me permet de récupérer les contrats d'un Client.

La table « Type_client » me permet de savoir à qu'elle catégorie appartient un Client (Particulier ou Entreprise).

La table « Type_contrat » me permet de savoir à quel volet de la start-up un contrat est destiné. Soit le volet Consulting soit le volet Business Plan

La table « Privilèges » me donne les fonctions et donc les privilèges de chaque administrateur.



```
CREATE TABLE "Client" (
  "id_client" serial NOT NULL,
  "nom_client" varchar(255) NOT NULL,
  "descriptif_client" varchar(255) NOT NULL,
  "representant" varchar(255) NOT NULL,
  "email_client" varchar(255) NOT NULL,
  "password_client" varchar(255) NOT NULL,
  "type_client" int NOT NULL,
  CONSTRAINT Client_pk PRIMARY KEY ("id_client")
) WITH (
  OIDS=FALSE
);

CREATE TABLE "Type_Client"(
  "id_type" int NOT NULL,
  "libelle" varchar(255) NOT NULL,
  CONSTRAINT Type_Client_pk PRIMARY KEY("id_type")
)WITH (
  OIDS=FALSE
);

CREATE TABLE "Rendezvous_prospect" (
  "id_rdv" serial NOT NULL,
  "nom_prospect"varchar(255) NOT NULL,
  "representant_prospect" varchar(255),
  "date_rdv" DATE NOT NULL,
  CONSTRAINT Rendezvous_prospect_pk PRIMARY KEY ("id_rdv")
) WITH (
  OIDS=FALSE
);
```

```
CREATE TABLE "Type_contrat" (
  "id_type" serial NOT NULL,
  "descriptif_type" varchar NOT NULL,
  CONSTRAINT Type_contrat_pk PRIMARY KEY ("id_type")
) WITH (
  OIDS=FALSE
);

CREATE TABLE "Contrat" (
  "id_contrat" serial NOT NULL,
  "num_client" serial NOT NULL,
  "type_contrat" serial NOT NULL,
  "liens_contrat" varchar UNIQUE,
  CONSTRAINT Contrat_pk PRIMARY KEY ("id_contrat")
) WITH (
  OIDS=FALSE
);

CREATE TABLE "Administrateurs"(
  "id_admin" serial NOT NULL,
  "nom_admin" varchar(255) NOT NULL,
  "prenom_admin" varchar(255) NOT NULL,
  "password_admin" varchar(255) NOT NULL,
  "numero_admin" varchar(255) NOT NULL,
  "email_admin" varchar(255) NOT NULL UNIQUE,
  "privilege_admin" serial NOT NULL,
  CONSTRAINT Admin_pk PRIMARY KEY ("id_admin")
) WITH (
    OIDS=FALSE
  );
```

```
CREATE TABLE "Privileges"(
  "id_privilege" serial NOT NULL,
  "libelle" varchar(255) NOT NULL,
  CONSTRAINT Privilege_pk PRIMARY KEY ("id_privilege")
) WITH (
    OIDS=FALSE
  );


ALTER TABLE "Contrat" ADD CONSTRAINT "Contrat_fk1" FOREIGN KEY ("type_contrat") REFERENCES "Type_contrat"("id_type") ON DELETE CASCADE;
ALTER TABLE "Client" ADD CONSTRAINT "Client_fk1" FOREIGN KEY ("type_client") REFERENCES "Type_Client"("id_type") ON DELETE CASCADE;
ALTER TABLE "Administrateurs" ADD CONSTRAINT "Admin_fk1" FOREIGN KEY ("privilege_admin") REFERENCES "Privileges"("id_privilege") ON DELETE CASCADE;
```

1.3 – Choix technique

Pour la base de données : j'utilise « Postgres ». Bien que « MySQL » beaucoup plus facile d'utilisation, « Postgres » offre une réelle plus-value dans le cas où cette application serait amenée à gérer un volume de données de plus en plus grand. Contrairement aux idées reçue elle n'est pas

beaucoup plus compliquée à utiliser que « MySQL » ou « Oracle » ,il y'a aussi un bonne documentation et une communauté très grande. L'installation des drivers, de « Pgadmin » (le gestionnaire graphique de Postgres » et l'utilisation de commande via la console est très facile.

1.3 - Helpers

Pour que cette modélisation de base de données soit plus réelle à mes yeux, j'ai utilisé JMerise qui est un bon outil pour faire des MCD et également DBdesigner qui s'inspire de l'UML pour faire des schémas de création de base de données.

2- Dans les entrailles : le Back-end

Nous rentrons au cœur du problème maintenant. Quels ont été mes choix de langages, d'hébergeurs……..nous allons le voir dans cette partie.

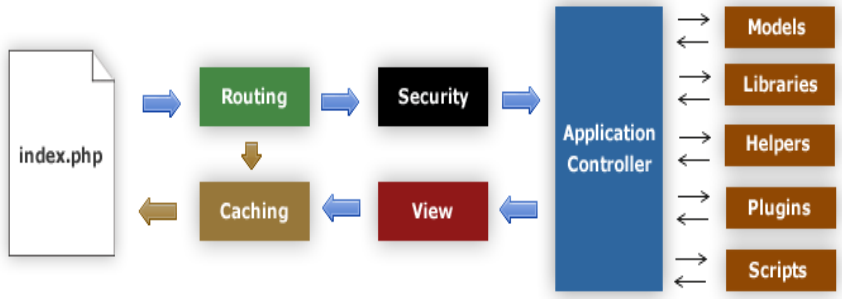
2.1- Choix de technologie

Pour le langage coté serveur j'ai utilisé le « PHP » qui est un langage impératif orienté objet. J'ai utilisé cette technologie car elle me permettait de réaliser mes besoins fonctionnels, c'est une technologie qui a une documentation très poussée (on peut faire beaucoup avec) et il y'a une communauté au niveau du PHP nombreuse mais qui tend à diminuer avec les nouvelles technologies qui apparaissent.

J'ai également eu recours à un **Framework PHP « CodeIgniter »** pour faciliter mes échanges avec la base de données, la gestion de la sécurité, le passage des paramètres de façon sécurisée, la mise sous forme de MVC (Model-View-Controller), la réécriture des URI et le routage de mes

différentes actions. J'ai utilisé « CodeIgniter » car il est facile à prendre en main au vue de la durée qui nous était impartie pour créer ce site et contrairement à des Framework comme Laravel, Lumen ou Symphony, on garde à peu près les règles syntaxiques du « PHP » de base.

Comment ce Framework fonctionne ?



- L'index : C'est le fichier qui sera toujours appelé en 1er par le navigateur

- Le routing : C'est lui qui récupérera l'URL et la décomposera en actions. Il a pour rôle de trouver le contrôleur à appeler.

- Security : Partie très importante, va sécuriser toutes les données entrantes : cookies, variable en post, get, put ou delete.

- C'est dans l'Application Controller que j'ai développé la plupart du code.

- Tous ce qui est en marron (model, helpers, plugin, Script et librairies) peuvent être appelé et ainsi j'ai pu utiliser l'ensemble des fonctions qu'elles fournissent notamment dans le « Helper » les fonctions de « Cookie ».

2.2- Sécurité

Ce volet est très important car de nos jours un site se doit d'être un minimum sécurisé notamment si l'on permet des authentifications dessus, qu'un Client ne puisse pas passer en administrateur et vice versa. Pour la sécurité de mon site j'ai utilisé un système de « Cookie ». J'utilise plusieurs « Cookies » selon le type d'utilisateurs. Dans ceux-ci je crypte avec un algorithme « sha256 », une clé secrète, une information qui est propre à l'utilisateur et également le moment auquel ce « Cookie » a été créé pour éviter que quelqu'un le prenne quelques jours après et puisses se logger impunément. Pour la création des « Cookies » j'utilise l' « Helper Cookie » fournie par « CodeIgniter ».

Egalement les mots de passes en base de données ne sont pas stockés en claire, j'utilise la fonction de hachage de mots de passes inhérente à « PHP » avec un algorithme « BCRYPT » qui est l'un des plus sûr actuellement dans ce langage.

Chaque fois qu'une action veut être menée par un utilisateur de l'application, on va regarder si toutes les informations qu'on a sur lui match bien avec le cookie haché dans le navigateur. Si oui il peut poursuivre son action, sinon on le déconnecte parce que c'est mal de modifier les Cookies et donc il n'a rien à faire là.

2.4 – Problèmes Rencontrés

L'utilisation de « CodeIgniter » malgré la documentation bien fournit était parfois difficile à comprendre. Notamment au niveau des requêtes pour la base de données.

3- Ce qui se voit : le Front-end

3.1- Choix de technologie

Niveau front-end, j'ai associé à « PHP » qui est en back-end de l' **« HTML »** qui est un langage de balisage et du **« JavaScript ».** J'ai choisi ces deux langages parce qu'ils sont très compatible avec le « PHP ». De plus l'association « PHP+HTML » est une association solide mais qui tend à disparaitre. Le « JavaScript » quand à lui m'a permis de rendre mon visuel un peu plus dynamique sans forcément recharger la page.

J'ai également utilisé un bout d' **« Angular »** pour un visuel : celui de la page d'accueil.

Niveau design et CSS j'ai utilisé le Framework CSS **« Semantic UI ».** Pourquoi celui-là et pas un autre ? Parce qu'au fil de mes recherches sur ce Framework je me suis rendu compte que de ceux auxquels j'ai pensé celui-là est le meilleur, niveau documentation, communauté, et esthétisme des composants et il est responsive. Pour cette start-up pour qui ce site doit servir de vitrine, j'ai trouvé très intéressant de se servir de « Semantic ».

3.2 – Difficultés rencontrées

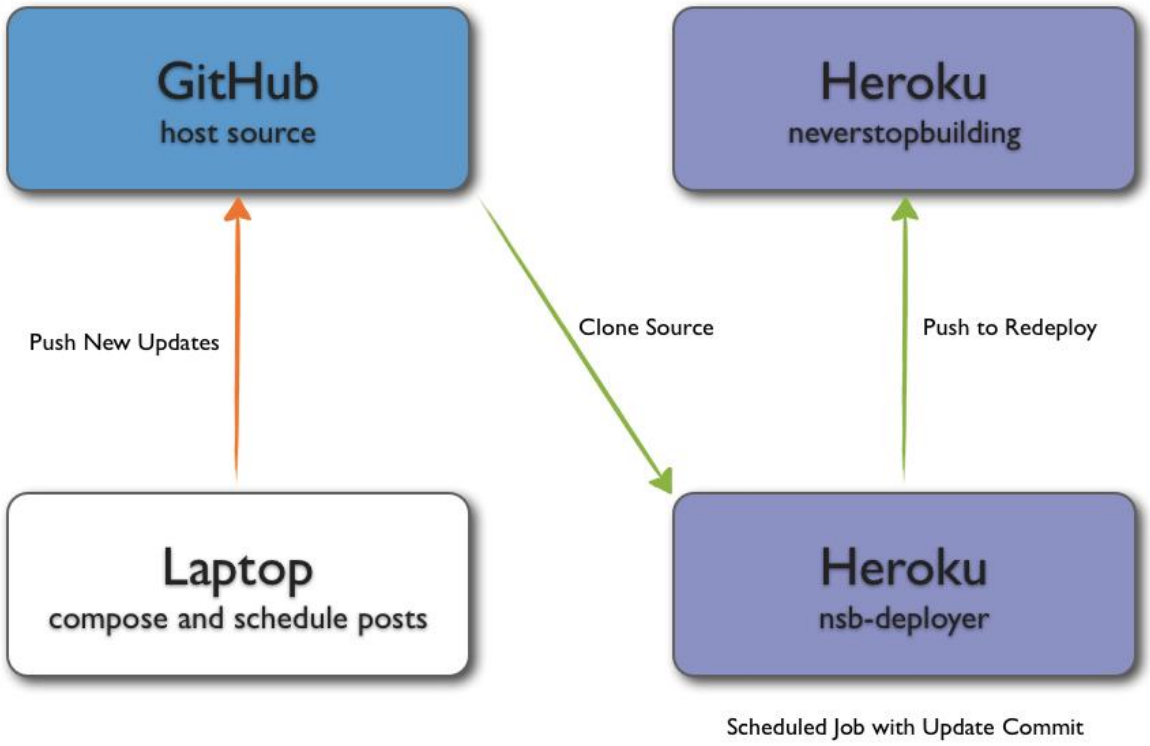
La plus grande difficulté ici a été d'installer Semantic avec toutes les dépendances que ça implique. Il a fallu passer par « Gulp » qui regroupe des outils pour installer des applications plus facilement. Installé « Node » également comme le suggère la documentation de « Semantic ». Ce n'était pas vraiment des difficultés mais plutôt de nouvelles approches qui ont retardé de quelques temps l'implantation du site.

Passons du coup à l'implantation de mon site en ligne. Dans la partie que va suivre j'expliquerais mon choix d'hébergeur et comment la communication se fait avec ce dernier.

4- Déploiement

J'ai choisi comme hébergeur « Heroku ». C'est une PaaS (Prestation as a Service) et comme sa dénomination l'indique il nous fournit plusieurs services. Les deux services qui m'ont particulièrement attirés ce sont : le fait qu'il puisse se connecter à mon repo « GitHub » et ensuite déployer de manière automatique le code qui s'y trouve. Ma base de données Postgres est également située sur l'un des serveurs d'Heroku. En tant qu'utilisateur de l'offre gratuite j'avais droit à 20 connexions simultanées et l'exécution de 10000 lignes dans le SGBD ce qui était plus que suffisant pour répondre à mes besoins fonctionnels de l'heure. Faire des backups de la base était simple et il s'adaptait très bien au fait que j'utilise « PHP ».

Ci-dessous vous trouverez un schéma sur la manière dont le déploiement s'effectue.



Comme dis plus haut dès qu'un nouveau commit est réalisé sur le mon Git pour ce projet « Heroku » se charge de le déployer automatiquement.

Conclusion

En conclusion, je peux dire que ce projet m'a été une nouvelle fois très bénéfique vu que je le fais pour la deuxième fois. Cette fois-ci je m'étais fixé l'objectif d'utiliser une nouvelle technologie et même si je réutilisais du « PHP » ne pas juste me limiter à lui et découvrir un Framework. Ici j'ai pu découvrir « CodeIgniter », ses avantages et ses désavantages.

Malgré tous je regrette de ne pas avoir pu utiliser des technologies comme Angular (qui nous a été présenté en cours) ou encore NodeJs (dont j'ai eu beaucoup d'écho positifs).

Des features à rajouter ? La modification des mots de passes des administrateurs, ajouter des outils financiers ou le cours de la bourse à l'application, ce serais très pratique pour les administrateurs, pour leur propre connaissance et pourquoi pas refaire le site plus tard en Angular avec du NodeJs pour le back-end.

J'ai eu beaucoup de plaisir à réaliser ce projet et j'espère que vous en aurez autant à le visiter.