23/10/2020

**Projet web MADERA**

Présentation de la maquette du module Configuration du projet MADERA

Alexy Auffret / Alan Gourdon

Table des matières

[Objectifs détaillés 3](#_Toc53996632)

[Dictionnaire de données et MCD de la base de données. 4](#_Toc53996633)

[La charte graphique 5](#_Toc53996634)

[Conclusion montrant les possibilités d'extension du projet, ses atouts (originalité), ses défauts et ses limites hypothétiques 6](#_Toc53996635)

[Extraits du code source commenté 7](#_Toc53996636)

# Objectifs détaillés

## Contexte

Afin de déterminer la faisabilité du projet et avoir une vision globale de l’application MADERA, la MOA à demandé de mettre en place sur une période d’une semaine une ébauche de l’application.

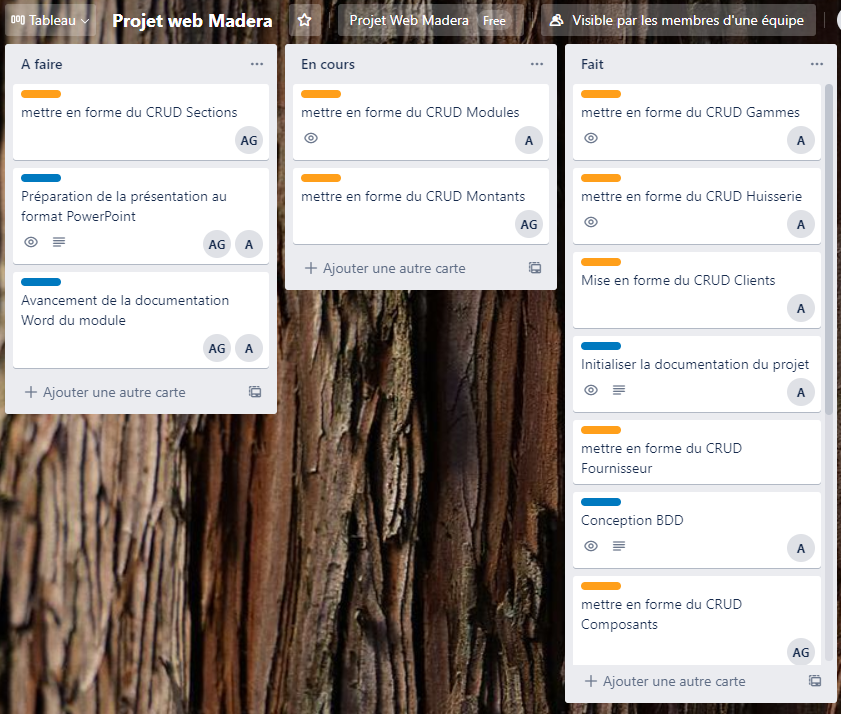
L’objectif de cette semaine est de développer l’ébauche de l’application web du projet en explorant une des pistes de développement déterminée par l’équipe projet MADERA.

## Modules à développer

Dans le cadre de cette démonstration, l’équipe de développement va se concentrer sur le développement du module « ***Configuration*** » dans un premier temps et par la suite sur le module « ***Conception de devis*** ». Dans la mesure ou le développement ne durera qu’une semaine il est fortement possible que le module « ***Conception de devis*** » ne soit pas développé dans sont intégralité, voir pas commencé, une rubrique de « *reste à faire »* sera présentée à la fin de ce document.

## Gestion de projet

Etant donné le court laps de temps et la taille de l’équipe il à été décidé de mettre en place un Trello afin de gérer les taches du projet.



## Technologies utilisées

Dans une optique d’optimisation du temps et de sécurisation de l’application le projet sera développé en **PHP** avec le Framework **Symfony 5** pour la partie Backend, en JavaScript avec le Framework VueJS pour la partie Frontend le design CSS serras géré par le Framework CSS Bootstrap 4.

**Installation du projet :**

* **WampServer V3.2.3**
* **PHP 7.4.9**
* **Apache 2.4**
* **Symfony 5**
* **Vuejs**
* **Composer**
* **Bootstrap Vue**
* **NPM**
* **mySql 5.7.31**

**Bundles Utilisés :**

* **Doctrine avec symfony/orm-pack (ORM du projet)**
* **Migration (permet à composer d’effectuer des migrations vers mySql)**
* **Webpack-encore (permet de gérer la relation entre Symfony et Vuejs )**
* **Form (Permet à composer de générer des formulaires et CRUD à partir des Entités)**
* **Maker (Permet de créer les entités, controllers, …)**

## Description du module Configuration

Sur la base des **Articles** saisis dans la « **Gestion de stock** », ce **Plugin** permettra principalement la conception des différents **Modules** de chaque **Gamme** afin de faciliter la réalisation de devis informatique.

La Gestion de stock sera le **Plugin** permettant la création des :

* Gammes
* Familles de composants
* Fournisseurs
* Composants (*et article*), chaque article sera attribué à un fournisseur et une famille de composant. En fonction de son utilisation dans la conception de gammes il pourra être attribué à plusieurs gammes.

### Configuration des gammes :

Les **gammes** sont différents types de Maisons Modulaires, elles varient en fonction de :

* La finition extérieure (bois, crépis…)
* Le type d’isolant utilisé (synthétique, naturel ou biologique)
* Le type de couverture (tuiles, ardoises…)
* La qualité des huisseries.

*On pourrait ainsi imaginer une gamme « premium » ou encore une gamme « classique ». etc.*

La configuration des **gammes** permettra la création des règles de calcul de chaque gamme en fonction du mode de conception de l’ossature bois :

* Sans angle
* Avec angles ouvrant
* Avec angles fermant…

### Configuration des Modules :

Ce **Plugin** permettra la conception des éléments de **Modules types :** Appelé **Modèles**

#### En Identifiant le module

* Attribuer le **Module** à une **Gamme**. Nommer le **Module** (*normaliser*)
* Sélectionner la Coupe de principe (*voir schémas annexes*)
* Sélectionner le CCTP (*caractéristiques de la dalle béton ou des plots béton recevant une lisse basse horizontale qui servira de support aux montants verticaux*)

#### En Sélectionnant les Composants du Module et en précisant les quantités par unité de base :

* Sections
* Lisses
* Contreforts
* Sabots d’assemblage
* Goujons de fixation
* Supports de sol
* Montants
* Départ – Sabots d’assemblage …
* Arrivée – Sabots d’assemblage …
* Intermédiaire – Sabots d’assemblage …
* Remplissage par unité entre deux montants
* Remplissages
* Visserie
* Panneaux
* …

Afin de faciliter la conception des **Modèles,** il sera possible de copier l’ensemble d’une configuration. Une fois un **Modèle** défini avec une pose à plat sur dalle, après copie on pourra créer le **Modèle** avec une pose sur plot en modifiant uniquement les supports de sol.

### Huisseries :

Les huisseries seront dépendantes de la **Gamme** du projet.

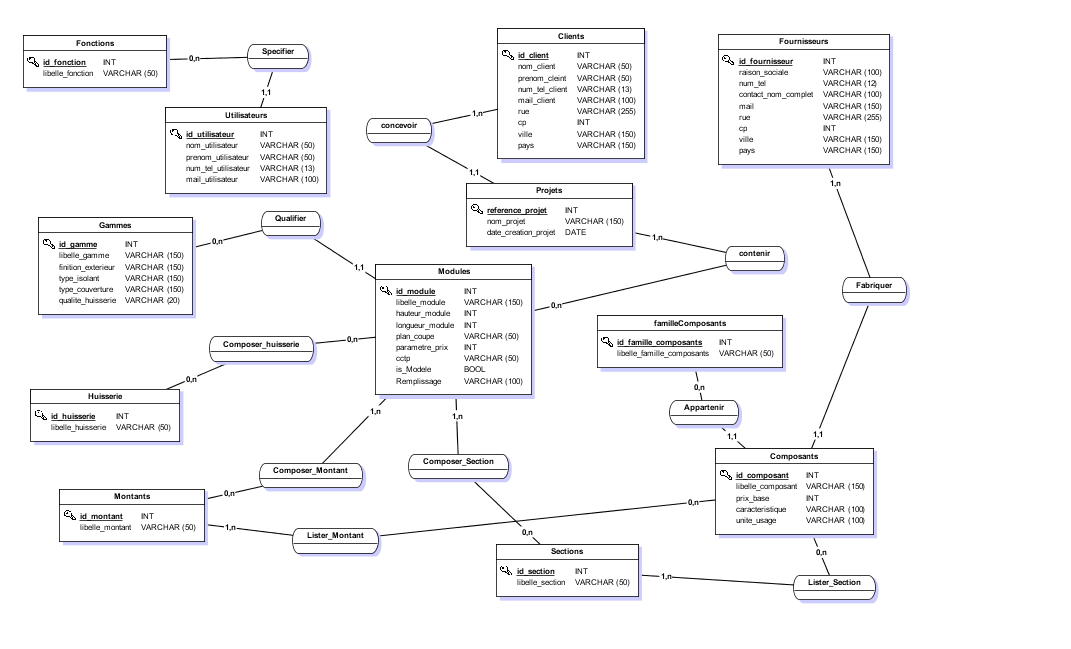
### Paramètres :

Permettra de :

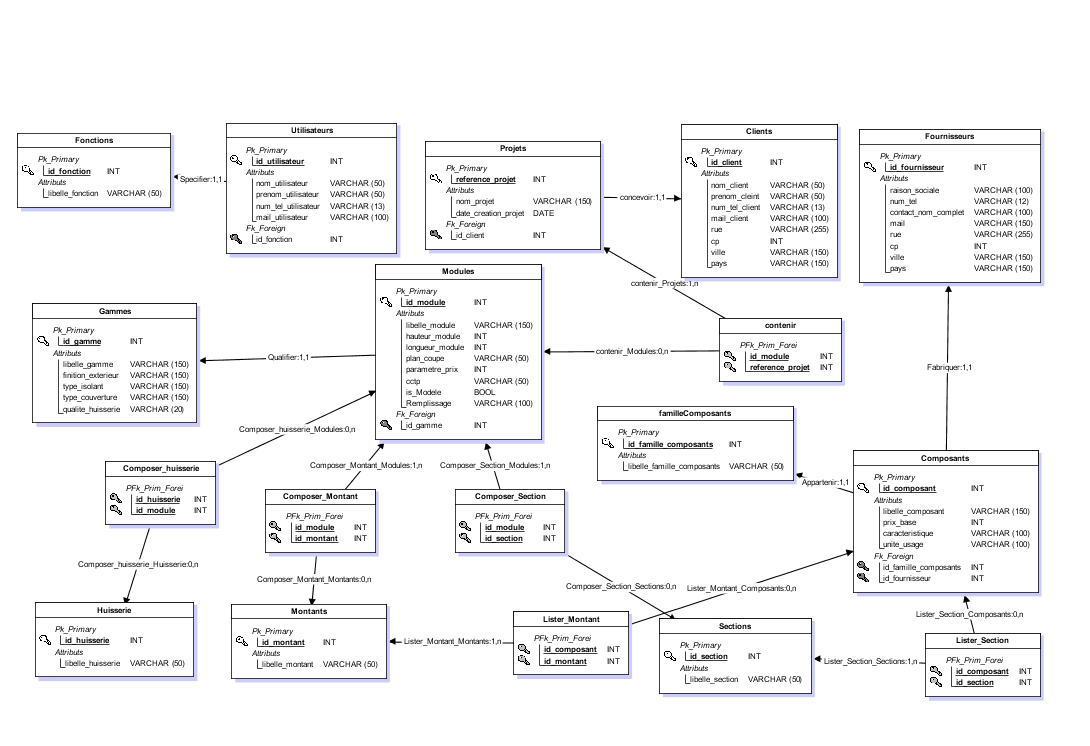
* Définir les pourcentages appliqués aux prix de base pour définir le prix de vente (*Marge entreprise - Marge commerciaux*)

# Dictionnaire de données et MCD de la base de données.

## MCD mis en place pour ce développement



## MLD mis en place pour ce développement



# La charte graphique

La charte graphique se base sur le Framework CSS Bootstrap 4 avec le logo et sur les couleurs du logo de l’entreprise MADERA.

# Conclusion

## Possibilités d'extension du projet

Il serait possible voir nécessaire d’ajouter un bundle d’authentification afin de limiter l’accès à certaine partie du logiciel en fonction des rôles / fonctions des utilisateurs, il est évidement nécessaire de développer les deux autres modules du logiciel qui sont les modules « Conception de devis » et « Modalités de paiement ».

## Atouts du projet

L’utilisation de Symfony à fortement accéléré la conception du Backend ainsi que l’utilisation de Bootstrap. Cela à permis à l’équipe de s’affranchir de taches redondantes (mise en place de charte graphique CSS ou création à la main de CRUD) et par extension de pouvoir se concentrer sur le contraintes métier du projet.

## Défauts et Limites hypothétiques

L’utilisation de Vuejs se limite au menu de l’application, nous aurions dut gérer le frontend sous forme d’API afin d’optimiser l’interaction de Vuejs avec Symfony et résoudre certains conflits rencontrés.

Nous aurions dû également utiliser la version de Symfony 4, plus stable et mon compliqué à mettre en place au niveau du paramétrage.

## Reste à faire

Dut à des soucis de configuration du projet notamment sur le paramétrage de Symfony 5, le développement du module Conception de devis n’a pas été commencé.

Il reste également à terminer le front des CRUD « Modules », « Sections » et « Montants ».

Le déploiement serveur n’as put être mis en place également par manque de temps.

# Extraits du code source commenté

## Extrait de code Symfony

### Exemple de code pour une Entité

<?php  
  
namespace App\Entity;  
  
use App\Repository\FournisseursRepository;  
use Doctrine\Common\Collections\ArrayCollection;  
use Doctrine\Common\Collections\Collection;  
use Doctrine\ORM\Mapping as ORM;  
  
*/\*\*  
 \** ***@ORM\Entity****(repositoryClass=FournisseursRepository::class)  
 \* Attributs de la classe fournisseur  
 \*/*class Fournisseurs  
{  
 */\*\*  
 \** ***@ORM\Id*** *\** ***@ORM\GeneratedValue*** *\** ***@ORM\Column****(type="integer")  
 \*/* private $id;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=100)  
 \*/* private $raison\_sociale;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=12)  
 \*/* private $num\_tel;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=100)  
 \*/* private $contact\_nom\_complet;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=150)  
 \*/* private $mail;  
  
 */\*\*  
 \** ***@ORM\OneToMany****(targetEntity=Composants::class, mappedBy="fournisseur")  
 \*/* private $composant;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=255)  
 \*/* private $rue;  
  
 */\*\*  
 \** ***@ORM\Column****(type="integer")  
 \*/* private $cp;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=150)  
 \*/* private $ville;  
  
 */\*\*  
 \** ***@ORM\Column****(type="string", length=255)  
 \*/* private $pays;

*/\*\*  
 \* Constructeur de la classe Fournisseur*

*\*/*  
 public function \_\_construct()  
 {  
 $this->composant = new ArrayCollection();  
 }

*/\*\*  
 \* Méthodes d’accès aux attributs de la classe Fournisseurs*

*\*/*  
 public function getId(): ?int  
 {  
 return $this->id;  
 }  
  
 public function getRaisonSociale(): ?string  
 {  
 return $this->raison\_sociale;  
 }  
  
 public function setRaisonSociale(string $raison\_sociale): self  
 {  
 $this->raison\_sociale = $raison\_sociale;  
  
 return $this;  
 }  
  
 public function getNumTel(): ?string  
 {  
 return $this->num\_tel;  
 }  
  
 public function setNumTel(string $num\_tel): self  
 {  
 $this->num\_tel = $num\_tel;  
  
 return $this;  
 }  
  
 public function getContactNomComplet(): ?string  
 {  
 return $this->contact\_nom\_complet;  
 }  
  
 public function setContactNomComplet(string $contact\_nom\_complet): self  
 {  
 $this->contact\_nom\_complet = $contact\_nom\_complet;  
  
 return $this;  
 }  
  
 public function getMail(): ?string  
 {  
 return $this->mail;  
 }  
  
 public function setMail(string $mail): self  
 {  
 $this->mail = $mail;  
  
 return $this;  
 }  
  
 */\*\*  
 \** ***@return*** *Collection|Composants[]*

*\* Méthodes d’accès à la liste des composants liés à ce fournisseur  
 \*/* public function getComposant(): Collection  
 {  
 return $this->composant;  
 }  
  
 public function addComposant(Composants $composant): self  
 {  
 if (!$this->composant->contains($composant)) {  
 $this->composant[] = $composant;  
 $composant->setFournisseur($this);  
 }  
  
 return $this;  
 }  
  
 public function removeComposant(Composants $composant): self  
 {  
 if ($this->composant->contains($composant)) {  
 $this->composant->removeElement($composant);  
 // set the owning side to null (unless already changed)  
 if ($composant->getFournisseur() === $this) {  
 $composant->setFournisseur(null);  
 }  
 }  
  
 return $this;  
 }  
  
 public function getRue(): ?string  
 {  
 return $this->rue;  
 }  
  
 public function setRue(string $rue): self  
 {  
 $this->rue = $rue;  
  
 return $this;  
 }  
  
 public function getCp(): ?int  
 {  
 return $this->cp;  
 }  
  
 public function setCp(int $cp): self  
 {  
 $this->cp = $cp;  
  
 return $this;  
 }  
  
 public function getVille(): ?string  
 {  
 return $this->ville;  
 }  
  
 public function setVille(string $ville): self  
 {  
 $this->ville = $ville;  
  
 return $this;  
 }  
  
 public function getPays(): ?string  
 {  
 return $this->pays;  
 }  
  
 public function setPays(string $pays): self  
 {  
 $this->pays = $pays;  
  
 return $this;  
 }  
}

### Exemple de code pour un Controller

<?php  
  
namespace App\Controller;  
*// imports des dépendances du controller*   
use App\Entity\Fournisseurs;  
use App\Form\FournisseursType;  
use App\Repository\FournisseursRepository;  
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;  
use Symfony\Component\HttpFoundation\Request;  
use Symfony\Component\HttpFoundation\Response;  
use Symfony\Component\Routing\Annotation\Route;  
  
*/\*\*  
 \** ***@Route****("/fournisseurs")*

*\* Route principale pour accéder au Controlleur Fournisseur  
 \*/*class FournisseursController extends AbstractController  
{  
 */\*\*  
 \** ***@Route****("/", name="fournisseurs\_index", methods={"GET"})*

*\* Route d’accès à l’index des fournisseurs qui renvoie la liste des  
 \* fournisseurs presents en bdd  
 \*/* public function index(FournisseursRepository $fournisseursRepository): Response  
 {  
 return $this->render('fournisseurs/index.html.twig', [  
 'fournisseurs' => $fournisseursRepository->findAll(),  
 ]);  
 }  
  
 */\*\*  
 \** ***@Route****("/new", name="fournisseurs\_new", methods={"GET","POST"})*

*\* Route de creation d’un nouveau fournisseur   
 \*/* public function new(Request $request): Response  
 {  
 $fournisseur = new Fournisseurs();  
 $form = $this->createForm(FournisseursType::class, $fournisseur);  
 $form->handleRequest($request);  
  
 if ($form->isSubmitted() && $form->isValid()) {  
 $entityManager = $this->getDoctrine()->getManager();  
 $entityManager->persist($fournisseur);  
 $entityManager->flush();  
  
 return $this->redirectToRoute('fournisseurs\_index');  
 }  
  
 return $this->render('fournisseurs/new.html.twig', [  
 'fournisseur' => $fournisseur,  
 'form' => $form->createView(),  
 ]);  
 }  
  
 */\*\*  
 \** ***@Route****("/{id}", name="fournisseurs\_show", methods={"GET"})*

*\* Route qui permet d’afficher la fiche d’un fournisseur en prenant en  
 \* parametre un fourisseur id  
 \*/* public function show(Fournisseurs $fournisseur): Response  
 {  
 return $this->render('fournisseurs/show.html.twig', [  
 'fournisseur' => $fournisseur,  
 ]);  
 }  
  
 */\*\*  
 \** ***@Route****("/{id}/edit", name="fournisseurs\_edit",   
 \* methods={"GET","POST"})*

*\* Route qui permet de modifier un fournisseur en prenant en parameter  
 \* un fourisseur id  
 \*/*public function edit(Request $request, Fournisseurs $fournisseur): Response  
 {  
 $form = $this->createForm(FournisseursType::class, $fournisseur);  
 $form->handleRequest($request);  
  
 if ($form->isSubmitted() && $form->isValid()) {  
 $this->getDoctrine()->getManager()->flush();  
  
 return $this->redirectToRoute('fournisseurs\_index');  
 }  
  
 return $this->render('fournisseurs/edit.html.twig', [  
 'fournisseur' => $fournisseur,  
 'form' => $form->createView(),  
 ]);  
 }  
  
 */\*\*  
 \** ***@Route****("/{id}", name="fournisseurs\_delete", methods={"DELETE"})  
 \* Route qui permet de supprimer un fournisseur en prenant en parameter  
 \* un fourisseur id  
 \*/* public function delete(Request $request, Fournisseurs $fournisseur): Response  
 {  
 if ($this->isCsrfTokenValid('delete'.$fournisseur->getId(), $request->request->get('\_token'))) {  
 $entityManager = $this->getDoctrine()->getManager();  
 $entityManager->remove($fournisseur);  
 $entityManager->flush();  
 }  
  
 return $this->redirectToRoute('fournisseurs\_index');  
 }  
}

### Exemple de code pour un formulaire

Dans le dossier madera/src/form se trouve les fichiers liés aux formulaires du projet

**Exemple de code pour le formulaire Composant :**

<?php  
  
namespace App\Form;  
  
use App\Entity\Composants;  
use App\Entity\FamilleComposants;  
use App\Entity\Fournisseurs;  
use App\Entity\Sections;  
use Symfony\Bridge\Doctrine\Form\Type\EntityType;  
use Symfony\Component\Form\AbstractType;  
use Symfony\Component\Form\Extension\Core\Type\NumberType;  
use Symfony\Component\Form\Extension\Core\Type\TextType;  
use Symfony\Component\Form\FormBuilderInterface;  
use Symfony\Component\OptionsResolver\OptionsResolver;  
  
  
class ComposantsType extends AbstractType  
{  
 public function buildForm(FormBuilderInterface $builder, array $options)  
 {  
 $builder

// Champ de type correspondant à un input de type Text pour  
 // l’attribut libelle\_composant de la classe Composants  
 ->add('libelle\_composant',TextType::class,[  
  
 'attr'=>[  
 'placeholder'=>"Libellé",  
 'class' => 'form-control mt-2'  
 ]  
 ])  
 ->add('caracteristique',TextType::class,[  
  
 'attr'=>[  
 'placeholder'=>"Caractéristique",  
 'class' => 'form-control mt-2'  
 ]  
 ])  
 ->add('unite\_usage',TextType::class,[  
  
 'attr'=>[  
 'placeholder'=>"Unité d'usage",  
 'class' => 'form-control mt-2'  
 ]  
 ])  
 ->add('prix\_base',NumberType::class,[  
  
 'attr'=>[  
 'placeholder'=>"Prix de base",  
 'class' => 'form-control mt-2'  
 ]  
 ])

// Champ correspondant à un input de type selectListe pour  
 // l’attribut fournisseur qui correspond à une entité du module  
 // de la classe Composants  
 ->add('fournisseur', EntityType::Class, array(  
 'class'=> Fournisseurs::class,  
 'choice\_label' =>'raison\_sociale',  
 'attr'=>[  
 'placeholder'=>"Raison Sociale",  
 'class' => 'form-control mt-2'  
 ]  
 ))  
 ->add('famille',EntityType::Class, array(  
 'class'=>FamilleComposants::class,  
 'choice\_label' => 'libelle\_famille\_composants',  
 'attr'=>[  
 'placeholder'=>"Famille",  
 'class' => 'form-control mt-2'  
 ]  
 ))  
 ;  
 }  
  
 public function configureOptions(OptionsResolver $resolver)  
 {  
 $resolver->setDefaults([  
 'data\_class' => Composants::class,  
 ]);  
 }  
}

## Extrait de code Twig

Dans le dossier madera/template se trouve les fichiers liés aux affichages twig du projet

**Exemple de code pour l’affichage du formulaire de création d’un Client :**

**Fichier Client/new.html.twig**

{% extends 'base.html.twig' %}  
  
{% block title %}New Clients{% endblock %}  
  
{% block body %}  
 <h1>Create new Clients</h1>  
  
 {{ include('clients/\_form.html.twig') }}  
  
 <a href="{{ path('clients\_index') }}">back to list</a>  
{% endblock %}

**Fichier Client/\_form.html.twig (Vue partielle intégrée dans Client/new.html.twig)**

{{ form\_start(form) }}  
 <div class="card-body">  
 <h2>Informations client</h2>  
 {{ form\_row(form.nom\_client, {'label':false}) }}  
 {{ form\_row(form.prenom\_client, {'label':false}) }}  
 {{ form\_row(form.num\_tel\_client, {'label':false}) }}  
 {{ form\_row(form.mail\_client, {'label':false}) }}  
 <h2>Adresse</h2>  
 {{ form\_row(form.rue, {'label':false}) }}  
 {{ form\_row(form.cp, {'label':false}) }}  
 {{ form\_row(form.ville, {'label':false}) }}  
 {{ form\_row(form.pays, {'label':false}) }}  
 </div>  
 <div class="card-footer bg-perso">  
 <button class="btn btn-success"><i class="fas fa-check"></i> {{ button\_label|default('Enregistrer') }}</button>  
 <a class="btn btn-secondary" href="{{ path('clients\_index') }}"> <i class="fas fa-arrow-left"></i> Retour</a>  
 </div>  
{{ form\_end(form) }}

## Extrait de code QueryBuilder

Dans le dossier madera/src/Repository se trouve les requêtes QueryBuilder pour les entités du projet.

Exemple de requête pour les Montants :

<?php  
  
namespace App\Repository;  
  
use App\Entity\Montants;  
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;  
use Doctrine\Persistence\ManagerRegistry;  
  
*/\*\*  
 \** ***@method*** *Montants|null find($id, $lockMode = null, $lockVersion = null)  
 \** ***@method*** *Montants|null findOneBy(array $criteria, array $orderBy = null)  
 \** ***@method*** *Montants[] findAll()  
 \** ***@method*** *Montants[] findBy(array $criteria, array $orderBy = null, $limit = null, $offset = null)  
 \*/*class MontantsRepository extends ServiceEntityRepository  
{  
 public function \_\_construct(ManagerRegistry $registry)  
 {  
 parent::*\_\_construct*($registry, Montants::class);  
 }  
  
 */\*\*  
 \** ***@return*** *Montants[] Retourne une collection de Montants filtrés sur les familles de composants  
 \*/* public function findByMontantsByComposantFamily($value)  
 {  
 return $this->createQueryBuilder('m')  
 ->andWhere('m.composants.famille\_id = :val')  
 ->setParameter('val', $value)  
 ->orderBy('m.id', 'ASC')  
 ->setMaxResults(10)  
 ->getQuery()  
 ->getResult();  
 }

## Extrait de migration

Lors de la création ou la modification d’Entités via composer il est nécessaire de faire une migration avec la commande « php bin/console make :migration » ce qui va généré une migration qui sera stockée dans le dossier « madera/migrations ».

Voici un exemple de code généré pour une migration :

<?php  
  
declare(strict\_types=1);  
  
namespace DoctrineMigrations;  
  
use Doctrine\DBAL\Schema\Schema;  
use Doctrine\Migrations\AbstractMigration;  
  
*/\*\*  
 \* Auto-generated Migration: Please modify to your needs!  
 \*/*final class Version20201022071125 extends AbstractMigration  
{  
 public function getDescription() : string  
 {  
 return '';  
 }  
  
 public function up(Schema $schema) : void  
 {  
 // Requête DQL générée pour mettre à jour la bdd à ârtir de la migration générée  
 $this->addSql('ALTER TABLE fournisseurs DROP FOREIGN KEY FK\_D3EF00414DE7DC5C');  
 $this->addSql('DROP TABLE adresses');  
 $this->addSql('DROP INDEX UNIQ\_D3EF00414DE7DC5C ON fournisseurs');  
 $this->addSql('ALTER TABLE fournisseurs DROP adresse\_id');  
 }  
  
 public function down(Schema $schema) : void  
 {  
 // Requête DQL afin d’annuler les dernières modifications de cette migration

$this->addSql('CREATE TABLE adresses (id INT AUTO\_INCREMENT NOT NULL, numero\_rue INT NOT NULL, rue VARCHAR(250) CHARACTER SET utf8mb4 NOT NULL COLLATE `utf8mb4\_unicode\_ci`, code\_postale INT NOT NULL, ville VARCHAR(250) CHARACTER SET utf8mb4 NOT NULL COLLATE `utf8mb4\_unicode\_ci`, PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8 COLLATE `utf8\_unicode\_ci` ENGINE = InnoDB COMMENT = \'\' ');  
 $this->addSql('ALTER TABLE fournisseurs ADD adresse\_id INT NOT NULL');  
 $this->addSql('ALTER TABLE fournisseurs ADD CONSTRAINT FK\_D3EF00414DE7DC5C FOREIGN KEY (adresse\_id) REFERENCES adresses (id)');  
 $this->addSql('CREATE UNIQUE INDEX UNIQ\_D3EF00414DE7DC5C ON fournisseurs (adresse\_id)');  
 }  
}

Il ne reste plus qu’à exécuter la commande « php bin/console doctrine :migrations :migrate » afin que doctrine exécute les migrations présentes dans le dossier et mette à jour la base de donnée du projet.

## Extrait de code Vuejs

Dans le dossier madera/assets se trouve les VuesJs du projet.

**Extrait de code vue du menu de l’application :**

Code du fichier Js controllant les composants VueJs :

import Vue from 'vue'  
import { ***BootstrapVue***, ***BootstrapVueIcons***, ***NavbarPlugin*** } from 'bootstrap-vue';  
  
//import 'bootstrap/dist/css/bootstrap.css';  
//import 'bootstrap-vue/dist/bootstrap-vue.css';  
  
Vue.use(***BootstrapVue***);  
Vue.use(***BootstrapVueIcons***);  
Vue.use(***NavbarPlugin***)  
  
import App from './js/App.vue'  
  
new Vue({  
 el: "#app",  
 components : {App},  
  
  
})

Code de la vue du composant VueJs pour le menu

<template>  
 <div>  
 <b-navbar toggleable="lg" type="dark" variant="success">  
 <img :src="'/img/logo\_madera.png'" alt="#" style="width: 20%">  
 <b-navbar-toggle target="nav-collapse"></b-navbar-toggle>  
  
 <b-collapse id="nav-collapse" is-nav>  
 <b-navbar-nav>  
 <b-nav-item href='/modules'>Modules</b-nav-item>  
 <b-nav-item href="/gammes" replace >Gammes</b-nav-item>  
 <b-nav-item href="/huisserie" >Huisseries</b-nav-item>  
 <b-nav-item href="/montants" >Montants</b-nav-item>  
 <b-nav-item href="/sections" >Sections</b-nav-item>  
 <b-nav-item href="/composants" >Composants</b-nav-item>  
 <b-nav-item href="/fournisseurs" >Fournisseurs</b-nav-item>  
 </b-navbar-nav>  
  
 </b-collapse>  
 </b-navbar>  
 </div>  
</template>  
  
<script>  
  
</script>  
  
<style>  
  
</style>