

# DOSSIER PROJET

**ONLINE**  
F O R M A P R O

- 1 - Introduction
- 2 - Liste des compétences du référentiel qui sont couvertes par le projet
- 3 - Résumé des projets
  - Stack de l'entreprise:
    - Outils utilisés
    - Technologies
- 4 - Spécifications techniques du projet
  - Projet Laravel
    - Langage PHP/HTML
    - Outils Laravel
      - Les principaux composants que j'ai utilisé sont:
        - Blade
        - Eloquent
        - SASS
        - Bootstrap
        - Telescope
  - Projet Wordpress
    - Outils Wordpress :
      - ACF
      - Gravity Forms
      - Gutenberg
- 6 - Réalisations du candidat (moi-même)
  - Front-end - Intégration d'une maquette pour un site wordpress
    - Réaliser une interface utilisateur web statique et adaptable
      - Mise en place, analyse du projet
      - Recherche et traduction de la documentation :
      - Mise en place du composant
  - Back-end - Travail sur Laravel en PHP
    - Travail commun sur tous les projets Laravel:
    - Travail sur le formulaire:
      - Petite précision sur le CSRF (ou aussi XSRF)
    - Travail sur le remplissage automatique de pdf
- 7 - Présentation du jeu d'essai
- 8 - Description de la veille
  - LinkedIn
  - Twitter
  - Youtube
  - Podcasts
  - Sites
- 9 - Description d'une situation de travail ayant nécessité une recherche
- 10 - Conclusion
  - Remerciements :

---

## 1 - Introduction

---

J'ai effectué un stage dans une agence web, DM Web, à Lons-le Saunier, en tant que stagiaire en développement web, du 10 Janvier au 4 mars 2022.

Durant ce stage, j'ai opéré sur différents projets, Wordpress d'un côté et avec le framework php Laravel de l'autre. J'ai fait de l'intégration de maquette sur wordpress, et des correction et ajouts de fonctionnalités en php sur des projets existants.

J'ai articulé mon dossier projet sur la partie la plus significative de mon stage en termes de technique : un projet Laravel d'application web type SaaS de gestion de personnel.

Je ne peux pas divulguer son nom ni de grandes parties du code. Cheminement de l'identification d'un problème, à sa résolution effective, il s'agit d'une fonctionnalité qui ajoute concrètement de la valeur à l'application.

Ensuite d'autres situations qui ont nécessité un développement particulier seront présentées, notamment un projet Wordpress.

## 2 - Liste des compétences du référentiel qui sont couvertes par le projet

---

- ☐ Maquetter une application
- ☒ Réaliser une interface utilisateur web statique et adaptable
- ☒ Développer une interface utilisateur web dynamique
- ☒ Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité
- ☒ Créer une base de données
- ☒ Développer les composants d'accès aux données
- ☒ Développer la partie back-end d'une application web ou web mobile
- ☐ Elaborer et mettre en oeuvre des composants dans une application de gestion de contenu ou e-commerce

---

## 3 - Résumé des projets

---

Durant mon stage, j'ai travaillé, entre autres, sur une application web de gestion de personnel.

C'est un projet récupéré d'une autre entreprise pour remettre au goût du jour l'interface, corriger des erreurs, et ajouter des fonctionnalités. J'y ai notamment développé une fonctionnalité de remplissage automatique de fichier PDF, un bouton d'inscription d'utilisateur automatique avec mot de passe aléatoire, une fonction de création de PDF.

Sur le projet Wordpress présenté, j'y ai fait l'intégration avec interface adaptable - responsive -, le développement de composants spécifiques avec une librairie.

### Stack de l'entreprise:

#### Outils utilisés

- Visual studio Code - Pour le code
- Mozilla Firefox - Pour les tests et le développement
- Google Chrome - Pour les tests et le développement
- Figma - Pour la lecture des maquettes
- Adobe XD - Pour la lecture des maquettes
- SourceTree - Pour la gestion des versions
- Laragon - Pour le serveur local php et les bases de données
- FileZilla - Pour l'envoi de fichiers au serveur
- Discord - Pour la communication et la gestion des projets
- Composer - pour l'installation de composants php

#### Technologies

- Wordpress
  - ACF
  - Gravity Forms
  - Template : Bootscore
- Laravel
  - Blade
  - Telescope
  - Tinker

## 4 - Spécifications techniques du projet

Le projet étant un passage de main d'une entreprise extérieure, nous n'avons pas de cahier de charges. Le projet est sur du Laravel 8 en PHP 7.4, la mise à jour des versions est prévue vers Laravel 9 et PHP 8. Cela ne devrait pas poser de problèmes dans l'immédiat, les versions antérieures continuant à être mises à jour au moins pour la sécurité. Voici les dates concernant le support des versions de PHP :

Branch	Initial Release		Active Support Until		Security Support Until	
<a href="#">7.4</a>	28 Nov 2019	2 years, 3 months ago	28 Nov 2021	3 months ago	28 Nov 2022	in 8 months
<a href="#">8.0</a>	26 Nov 2020	1 year, 3 months ago	26 Nov 2022	in 8 months	26 Nov 2023	in 1 year, 8 months
<a href="#">8.1</a>	25 Nov 2021	3 months ago	25 Nov 2023	in 1 year, 8 months	25 Nov 2024	in 2 years, 8 months

et les projets Laravel aussi auront besoin d'une mise à jour pour que la sécurité soit optimale:

Version	Released	Active Support	Security Support	Release
9.x	4 weeks ago (08 Feb 2022)	Ends in 1 year and 5 months (08 Aug 2023)	Ends in 1 year and 11 months (08 Feb 2024)	<a href="#">9.4.1</a>
8.x	1 year and 6 months ago (08 Sep 2020)	Ends in 4 months and 2 weeks (26 Jul 2022)	Ends in 10 months (24 Jan 2023)	<a href="#">8.83.2</a>

## Projet Laravel

### Langage PHP/HTML

### Outils Laravel

Laravel est un Framework PHP permettant de créer une application web, du simple blog à l'application métier complexe. Il permet une évolutivité et promet une mise en œuvre simplifiée dans toutes les couches de développement. Laravel respecte les principes du MVC (Modèle, Vue, Contrôleur)

Laravel est basé sur le framework Symfony, dont il reprend quelques onze composants. Les différences fondamentales entre Laravel et Symfony sont essentiellement la localisation des projets. Symfony est essentiellement utilisé en France de par son origine française, Laravel, issu de Symfony est plus utilisé dans le monde. Le choix se fera donc en fonction de la portée nationale ou internationale des projets.

A savoir aussi que Laravel est plus permissif donc plus accessible aux débutants, mais est, par conséquent, potentiellement moins facilement maintenable.

### Les principaux composants que j'ai utilisé sont:

#### Blade

Blade est un moteur de template, il va nous aider à mettre en page les vues, faire presque tout ce que nous permet php, en incorporant des fonctionnalités de sécurité. Il permet d'afficher des variables, de créer des conditions, des boucles avec compteur, une sécurisation de formulaires, etc.

Il m'a servi à faire des conditions, à insérer des données dans les vues, faire des liens entre les pages ainsi que des formulaires.

## Eloquent

Eloquent est un ORM (Object-Relational Mapper) c'est un outil qui permet d'interagir facilement avec la base de données, un CRUD facilité, chaque table de la base de données a un modèle Eloquent associé, ce qui permet une bonne visibilité, et une sécurité facilement accessible (Les modèles comportent des "fillable", les données autorisées à être écrites par l'utilisateur dans un formulaire par exemple.)

Eloquent a été utile pour toutes mes requêtes sql, sécurisées et simplifiées.

## SASS

Sass est un langage à compiler permettant d'écrire du CSS de manière plus générale. On peut utiliser des variables, des conditions, cela permet de créer des feuilles de style plus complexes plus facilement. C'est néanmoins un préprocesseur qui nécessite d'être compilé pour rendre un fichier en .CSS utilisable par le navigateur

J'ai pu ajuster des couleurs simplement en changeant une variable, ainsi que le style de certains éléments sans toucher au css qui était minifié.

## Bootstrap

Bootstrap est une librairie CSS et Javascript (en option) permettant de faire des règles CSS par le biais des classes HTML. Comme pour Tailwind, on définit les règles en nommant des classes par exemple "mx-5" qui crée des marges à gauche et à droite de 5rem, équivalent à :

```
{ margin : auto 5rem;}
```

Bootstrap marche bien en synergie avec Sass. Sass pour établir les règles générales et les objets spécifiques, Bootstrap pour la standardisation des composants et les ajustements dans les vues.

Pour du placement de composants, Bootstrap m'a fait gagné beaucoup de temps. Sur des objets spécifiques et des placements standards il est plus simple de lui donner une classe "mx-auto" plutôt que donner un nom spécifique, et ajouter les règles dans le fichier Sass.

## Telescope

Laravel Telescope est un outil permettant d'avoir un aperçu de toutes les requêtes entrantes et sortantes, les erreurs etc. Très pratique pour connaître la raison d'une erreur, ou optimiser une requête.

Telescope m'a été utile dans un projet où j'avais pour tâche de corriger une erreur 500.

# Projet Wordpress

Langage PHP/HTML/CSS

## Outils Wordpress :

### ACF

ACF ou Advanced Custom Fields est un plugin WordPress permettant de faire des champs personnalisés à l'endroit où l'on veut, et administrer son contenu depuis la page d'administration wordpress.

### **Gravity Forms**

Gravity Forms est un plugin dédié aux formulaires. Il permet la création, le suivi et l'administration d'un formulaire très simplement. On peut ajouter des captcha, faire des transactions et évidemment toutes les fonctions qu'un formulaire peut faire.

### **Gutenberg**

C'est l'éditeur par défaut de Wordpress. Il est basé sur le principe de blocs, on peut faire de la mise en page facilement, et le tout reste administrable. C'est une interface logicielle permettant de se passer du html

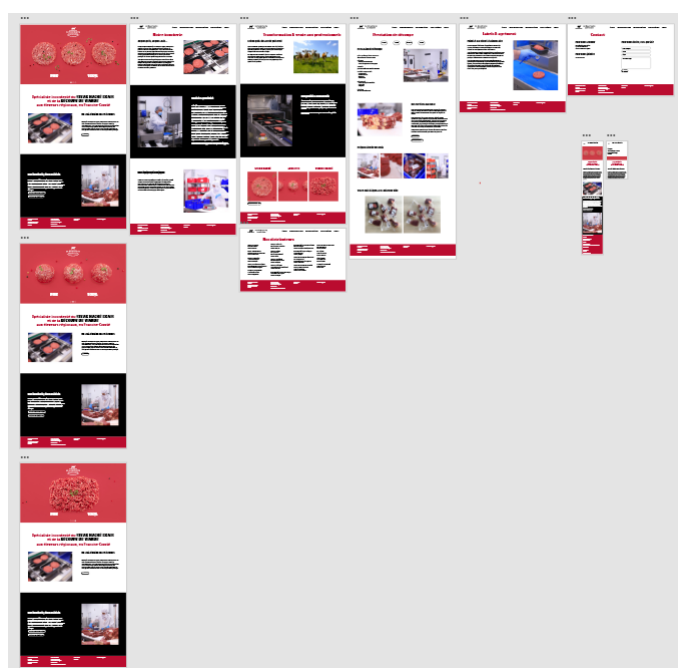
## 6 - Réalisations du candidat (moi-même)

### Front-end - Intégration d'une maquette pour un site wordpress

#### Réaliser une interface utilisateur web statique et adaptable

##### Mise en place, analyse du projet

J'ai réalisé l'intégration d'une maquette sur Wordpress, j'ai installé une instance de wordpress en local, puis envoyé le tout sur le serveur, ensuite mon tuteur a installé les plugins (ACF, Gravity Forms, WPS Hide Login, Duplicator, Yoast SEO) et le thème parent à modifier, Bootscore. Le thème Bootscore est un thème un peu spécial. "starter theme" il sert principalement à avoir un cadre, faire rapidement un agencement à l'aide de Bootstrap et est conçu pour être modifié. J'ai pris connaissance de la maquette:



Le site se découpe en 9 pages

- Accueil
- Contact
- Labels & agrément
- Mentions légales
- Nos distributeurs
- Notre boucherie
- Politique de confidentialité
- Prestation de découpe
- Transformation & vente aux professionnels

Le site étant un site vitrine il n'y aura pas d'articles, pas de e-commerce. La seule interaction se fera via le formulaire de contact.

Il y a essentiellement 3 types de pages:

- L'alternance noir/blanc



- Les pages classiques alternées gauche/droite
- La page contact

Le développement s'est fait sur le thème enfant d'un thème "générique", Bootscore. C'est un thème axé autour de Bootstrap qui permet une assez grande souplesse dans le design. Il a un menu inclus et quelques facilités.

Le principe du thème enfant est que tous les fichiers du thème parent sont pris en compte, seulement, les fichiers du thème enfants remplacent ceux du thème parent. Cela permet de créer des "sous thèmes" sans modifier l'original.

**Particularité** : il y a un slider en page d'accueil, dont nous aborderons le développement.

### Recherche et traduction de la documentation :

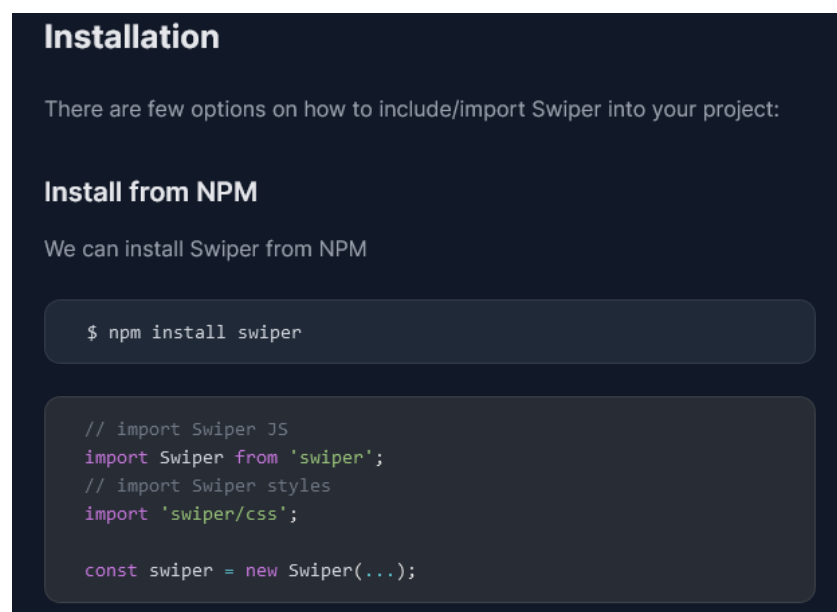
Sur la maquette je devais intégrer un carrousel d'images.

J'hésitais entre le carrousel Bootstrap classique et un carrousel un peu plus spécifique avec la librairie SwiperJs . J'ai opté pour SwiperJs par revanche car je n'avais pas su m'en servir plus tôt dans la formation.

### La librairie est en anglais:

Il y a plusieurs options pour inclure ou importer SwiperJs dans votre projet:

On peut installer SwiperJs avec NPM:



Par défaut, SwiperJs exporte seulement l'essentiel sans les modules additionnels (comme Navigation, Pagination, etc.). Il faut donc les importer et les configurer aussi.

By default Swiper exports only core version without additional modules (like Navigation, Pagination, etc.). So you need to import and configure them too:

```
// core version + navigation, pagination modules:
import Swiper, { Navigation, Pagination } from 'swiper';
// import Swiper and modules styles
import 'swiper/css';
import 'swiper/css/navigation';
import 'swiper/css/pagination';

// init Swiper:
const swiper = new Swiper('.swiper', {
  // configure Swiper to use modules
  modules: [Navigation, Pagination],
  ...
});
```

Si vous voulez importer SwiperJs avec tous les modules alors il doit être importé à partir de `swiper/bundle`

If you want to import Swiper with all modules (bundle) then it should be imported from `swiper/bundle` :

```
// import Swiper bundle with all modules installed
import Swiper from 'swiper/bundle';

// import styles bundle
import 'swiper/css/bundle';

// init Swiper:
const swiper = new Swiper(...);
```

Utiliser SwiperJs avec un CDN

Si vous ne voulez pas inclure les fichiers SwiperJs dans votre projet, vous pouvez l'utiliser via un CDN. Les fichiers suivants sont disponibles:

### Use Swiper from CDN

If you don't want to include Swiper files in your project, you may use it from CDN. The following files are available:

```
<link
  rel="stylesheet"
  href="https://unpkg.com/swiper@8/swiper-bundle.min.css"
/>

<script src="https://unpkg.com/swiper@8/swiper-bundle.min.js"></script>
```

Si vous utilisez un module ES dans un navigateur, il y a une version CDN pour ça aussi:

Si vous voulez utiliser les fichiers \*SwiperJs\*, vous pouvez les télécharger directement sur <https://unpkg.com/swiper@8/>

If you use ES modules in browser, there is a CDN version for that too:

```
<script type="module">
  import Swiper from 'https://unpkg.com/swiper@8/swiper-bundle.esm.browser.min.js'

  const swiper = new Swiper(...)
</script>
```

### Download assets

If you want to use Swiper assets locally, you can directly download them from <https://unpkg.com/swiper@8/>

## Mise en place du composant

Désormais nous allons travailler seulement dans le thème enfant.

J'ai choisi le bundle CDN pour avoir accès à la pagination et à la navigation, sans rajouter de fichiers au reste du thème. J'ai ajouté la ligne suivante dans le fichier "functions.php".

```
// swiper.js
wp_enqueue_script('swiper', 'https://unpkg.com/swiper/swiper-bundle.min.js', null, null, false);
```

j'ai créé le fichier "page-full-width-image.php" dans le dossier du thème enfant:

```
/**
 * Template Name : Full Width Image with carousel
 *
 * @link https://developer.wordpress.org/themes/basics/template-hierarchy/
 *
 * @package Bootscore
 */

get_header();
wp_enqueue_script('swiper');
?>

<header>
  <div class="swiper mySwiper">
    <!-- Texte et logo -->
    <?php
      if (get_field('logo_banniere')):
        ?>
        <div class="logo-banniere">
          <?= get_field('logo_banniere') ?>
        </div>
        <?php
      endif
    ?>
    <div class="text-banniere">
      <div><?= get_field('texte_banniere_g') ?></div>
      <div><?= get_field('texte_banniere_d') ?></div>
    </div>
    <div class="swiper-wrapper">
      <?php
      if (have_rows('carousel_group_img')) :
```

```

wp_enqueue_style('swiper-css');
while (have_rows('carousel_group_img')) : the_row();
?>
<div class="swiper-slide">
</div>
<?php
    endwhile;
endif;
?>
</div>
<div class="swiper-pagination"></div>
</div>
</header>
<div id="content" class="site-content">
<div id="primary" class="content-area">
    <?php the_content(); ?>
</div><!-- #primary -->
</div><!-- #content -->
<?php
get_footer();

```

### Un peu d'explications:

```

/**
 * Template Name : Full width Image with carousel
 *
 * @link https://developer.wordpress.org/themes/basics/template-hierarchy/
 *
 * @package Bootscore
 */

```

Template Name : Cela va rendre disponible le choix du modèle de page à sa création, ou à sa modification. Cela permet de faire des modèles génériques pour des pages un peu plus spécifiques. Dans cet exemple, on s'occupe de la page d'accueil qui contient un slider, les autres n'en ont pas, elles ont donc une structure différente. La page par défaut s'appelle *page.php*.

```
wp_enqueue_script('swiper');
```




n'est pas obligatoire et totalement facultatif. Il est le script est déjà déclaré et chargé dans le fichier *functions.php*.

```
get_field(quelquechose)
```

C'est une fonction appartenant à un plugin nommé ACF (Advanced Custom Fields) elle sert à faire le lien entre des champs créés dans l'administration wordpress et les pages, cela permet de faire comme ici, des modules personnalisables. Ici on peut changer le texte et les images du diaporama facilement.

## Bannière

### Images du carousel

	carousel_img	
1		
2		
3		

Ajouter un élément

```
<div class="swiper mySwiper">
<div class="swiper-wrapper">
```

Ce sont des classes spécifiques à SwiperJs pour insérer un carousel.

```
if (have_rows('carousel_group_img')) :
wp_enqueue_style('swiper-css');
while (have_rows('carousel_group_img')) : the_row();
```

Là on a des fonctions propres à wordpress : **have\_rows** va faire un simple check. Si son attribut existe, la fonction va retourner true, sinon elle retourne false. Elle est utilisée avec **the\_row()** que l'on verra par la suite.

`wp_enqueue_style` permet, comme `wp_enqueue_script` d'insérer un fichier dans la boucle d'exécution de wordpress. En l'occurrence le fichier CSS de \*SwiperJs\*.

Et pour finir, la boucle `while` conjuguée aux fonctions **have\_rows()** et **the\_row()**. `the_row()` va lister sous forme de tableau associatif toutes les valeurs au format `name => value`.

Ce bout de code signifie : s'il y a du contenu dans `carousel_group_img`, il sera affiché avec le style du fichier `swiper-css`.

Sinon, rien ne s'affiche et il n'y aura pas d'erreur qui bloquerait l'exécution du site.

Pour que le diaporama soit responsive et cohérent avec la maquette, j'ai ajouté quelques règles CSS:

```

/* BANNIERE */
.swiper {
  width : 100vw; /*Je veux que la largeur totale soit de 100% viewport, c'est
à dire toute la largeur de l'écran */
  height : auto;
  z-index : 1031; /*Z-index correspond à l'indice de superposition des
éléments. Plus le chiffre est grand, plus l'élément est devant */
}
.swiper-slide {
  text-align : center;
  font-size : 18px;
  background : #D03E49; /*C'est la couleur que l'on va voir sur les côtés des
images. Ne voulant pas déformer les images, ni faire un fond qui se détache trop,
j'ai choisi une couleur moyenne du fond des images */

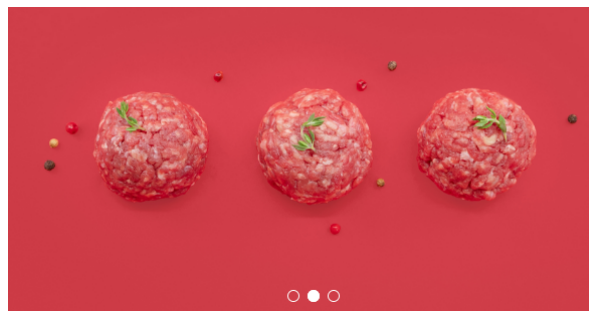
  /* J'ai vraiment voulu que le texte soit centré verticalement */
  display : -webkit-box;
  display : -ms-flexbox;
  display : -webkit-flex;
  display : flex;
  -webkit-box-pack : center;
  -ms-flex-pack : center;
  -webkit-justify-content : center;
  justify-content : center;
  -webkit-box-align : center;
  -ms-flex-align : center;
  -webkit-align-items : center;
  align-items : center;
}
.swiper-slide img {
  display : block;
  max-width : 1920px; /*La largeur maximum de l'image du carousel, pour éviter
qu'elle soit coupée ni déformée */
  width : auto;
  max-height : 1060px;
  height : 98vh; /*Hauteur maximale de 98% de l'écran, pour des raisons
d'ergonomie j'ai voulu que l'on voit que l'on peut scroller */
  object-fit : cover; /*En cas de petits écrans, cela permet de ne pas
déformer l'image et de la rogner de manière centrée */
  z-index : 1031;
}
.swiper-pagination-bullets.swiper-pagination-horizontal{
  width : 100vw;
}
.swiper-pagination-bullet { /*Les pagination bullets sont les points blancs en
bas du carousel */
  width : 25px;
  height : 25px;
  border : 3px solid white;
  box-sizing : border-box;
  background-color : rgba(255, 255, 255, 0);
  opacity : 1;
}
.swiper-pagination-bullet-active {

```

```

    background-color : white;
}
.logo-banniere{ /* Le logo du site */
    position:absolute; /*En position absolue, le point qui fait référence pour
le placement est le premier pixel en haut à gauche */
    top : 50px;
    left : calc(50vw - 127px); /*On va calculer le milieu de l'écran, moins la
moitié de la largeur du logo */
    z-index : 1032;
}
.text-banniere{
    max-width : 1920px;
    width : 100vw;
    left : 50%;
    transform : translate(-50%, -50%); /*On va traduire le texte de 50% de sa
largeur vers la gauche et 50% de sa hauteur vers le bas */
    position : absolute;
    bottom : 150px;
    display : flex; /* La disposition du texte va s'adapter selon la longueur du
texte */
    justify-content : space-around;
    font-family : 'Antique Olive Std Compact', Arial, Helvetica, sans-serif;
    color : white;
    font-size : 66px;
    z-index : 1032;
}
/* END BANNIERE */
/* RESPONSIVE BANNIERE */
@media screen and (max-width : 1600px) { /*En dessous de 1600px de large */
    .swiper-slide img{
        width : 100vw;
        height : auto;
    }
}
@media screen and (max-width : 1080px) , (max-height : 600px) {
    .text-banniere{
        display : none; /* en dessous de 1080px de large et/ou en dessous de
600px de hauteur, on enlève le texte */
    }
    .swiper-pagination-bullet{
        width : 12px;
        height : 12px;
        border : 1px solid white;
    }
}
@media screen and (max-width : 800px) {
    .logo-banniere{
        display : none; /* En dessous de 800px on enlève le logo, les autres
règles s'appliquent car elles n'ont pas de limite minimum */
    }
    .swiper-slide img{
        width : 100vw;
        height : auto;
    }
}
/* END RESPONSIVE BANNIERE */

```



On peut voir que sur grand écran jusqu'au téléphone, le slider s'adapte correctement.



## Back-end - Travail sur Laravel en PHP

### Travail commun sur tous les projets Laravel:

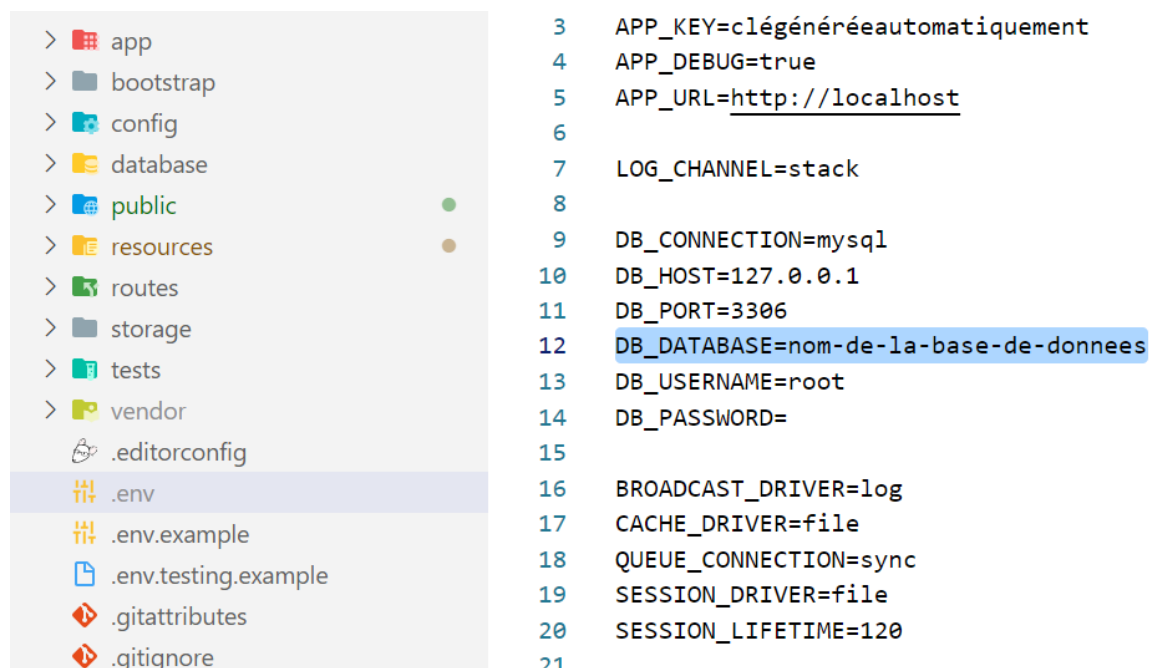
Rapatriement des données via la commande `git clone <url du repository>`

Installation de Laravel et des composants via la commande `composer install`

Création une copie du fichier '.env.example' nommée '.env'. dans la partie DATABASE on va spécifier les variables d'accès à la base de données et lui donner un nom.

Générer la clé d'encryption avec `php artisan key:generate`

Ensuite on va créer une base de données pour importer une éventuelle base de données existante, on va reprendre le nom inscrit dans le fichier '.env'.



The image shows a file explorer on the left with the following structure:

- > app
- > bootstrap
- > config
- > database
- > public
- > resources
- > routes
- > storage
- > tests
- > vendor
- .editorconfig
- .env (selected)
- .env.example
- .env.testing.example
- .gitattributes
- .gitignore

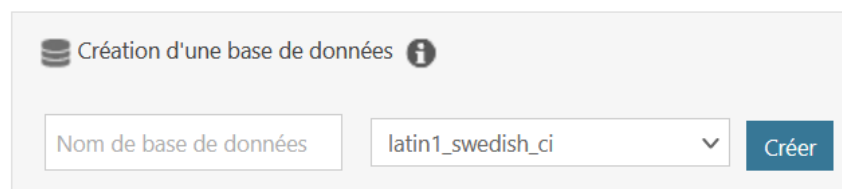
On the right, the content of the .env file is shown, with line numbers 3 through 21:

```

3 APP_KEY=clégénéréeautomatiquement
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=nom-de-la-base-de-donnees
13 DB_USERNAME=root
14 DB_PASSWORD=
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
21

```

Si on importe une base de données, on va importer le fichier de la base avec l'outil de gestion de BDD (attention aux limites de poids de Base de données).



The image shows a form titled "Création d'une base de données" with an information icon. It contains a text input field labeled "Nom de base de données" with the value "latin1\_swedish\_ci" and a dropdown menu showing "latin1\_swedish\_ci". A blue "Créer" button is to the right.

S'il n'y a pas de base de données, on va faire la commande `php artisan migrate`. qui va exécuter tous les fichiers contenus dans le dossier database/migrations

Ensuite, si un fichier seed est présent - des placeholder dans la base de données en exemple - on peut faire la commande `php artisan db:seed`.

Ensuite, si on a un logiciel de création d'environnement de développement local, type Laragon ou Xampp, il y a juste à ajouter le projet à l'environnement. Sinon on peut lancer le serveur PHP avec la commande `php artisan serve` mais il faudra tout de même de lancer le serveur de base de données.

## Travail sur le formulaire:

J'avais pour tâche d'améliorer le formulaire de déclaration d'accident de travail en automatisant le remplissage du formulaire le plus possible. J'ai donc automatisé le remplissage de la partie salarié puisqu'on l'avait déjà. L'objectif est de remplir des champs à la sélection du salarié, sans rechargement de page si possible.

La réflexion est la suivante : créer un formulaire avec la liste des employés qui va appeler un script au clic pour remplir les informations nécessaires.

Petite précision, pas besoin de sécuriser ce formulaire plus que ça puisqu'on ne peut pas entrer de données sur la base de données avec ce formulaire. Les autres par contre nécessitent un token csrf (voir plus bas).

```
<form action="{{ route('customer.accidents.create') }}" method="GET">
  <div class="row">
    <div class="col-12 mb-3">
      <div class="form-group">
        <label for="personnel_id" class="form-label">Employé <span
class="text-danger">*</span></label>
        <select onchange="selectEmployee()" name="employee_id"
id="employee_id" class="form-control select2" required>
          @foreach ($users as $user)
            <option value="{{ $user->id }}">{{ strtoupper($user->nom). '
' . $user->prenom }}</option>
          @endforeach
        </select>
      </div>
    </div>
  </div>
</form>
```

Un peu d'explications :

les doubles accolades {{ }} permettent d'interagir avec le moteur Blade

`route('customer.accidents.create')` correspond à la route définie qui comme on peut le voir ici dans le fichier web.php:

```
{Route::post('/accidents/{company}/create', 'Admin\AccidentController@create')-
>name('create');}
```

On appelle la méthode 'create' dans le contrôleur 'AccidentController',  
l'url est 'nom du site'/accidents/nom de l'entreprise/create  
et le nom de la route est 'create'

La route du formulaire va donc faire appel à la méthode create du contrôleur AccidentController. Nous verrons les contrôleurs un peu plus tard.

Les classes des balises sont des classes Bootstrap permettant de styliser le formulaire. On va faire une boucle @foreach qui est une fonction de Blade qui fait exactement la même chose qu'un foreach en PHP natif, c'est à dire faire une boucle jusqu'à ce qu'il n'y ait plus d'éléments sur lesquels boucler, comme si on faisait `if(i=0; i < user.length; i++)`.

```
@foreach ($users as $user)
  <option value="{{ $user->id }}">{{ strtoupper($user->nom). ' ' . $user->prenom }}
</option>
@endforeach
```

Ici la variable `$users` est appelée. Elle est donnée par la méthode `create` du contrôleur `AccidentController` qui affiche la page du formulaire.  
 Pour insérer une variable dans une vue on l'appelle avec une requête de base de données via Eloquent et retourne la vue correspondante avec les données:

`AccidentController`

```
public function create(Request $request)
{
    $auth = Auth::user();
    //On envoie une requête pour récupérer l'authentification

    if (empty($auth)) {
        abort(404);
    } //Si l'utilisateur n'est pas authentifié, on envoie une page 404

    $entreprise = $auth->entreprise;
    //on envoie une requete pour récupérer l'entreprise concernée

    if (empty($entreprise)) {
        abort(404);
    } //Si l'entreprise n'existe pas, on envoie une page 404

    $users = $entreprise->staff;
    //on envoie une requete pour récupérer le salarié concerné

    if (empty($users)) {
        $request->session()->flash('warning', 'Aucun employé enregistré dans
l\'entreprise');
    } //Si le salarié n'existe pas, on envoie un bandeau disant qu'il n'y a
pas d'employé

    return view('customer.accidents.create', compact('entreprise',
'users'));
    //on retourne la vue "accidents/create" avec les variables entreprise et
users
}
```

Sur cette vue, on peut donc récupérer des données des objets `$entreprise`, et `$users`.

```
<label for="personnel_id" class="form-label">Employé <span class="text-danger">*
</span></label>
<select onchange="selectEmployee()" name="employee_id" id="employee_id"
class="form-control select2" required>
    @foreach ($users as $user)
        <option value="{{ $user->id }}">{{ strtoupper($user->nom).' '.$user->
prénom }}</option>
    @endforeach
</select>
```

On récupère donc un par un tous les noms et prénoms des salariés avec comme valeur les Id des salariés. On les met en majuscule et ils viennent alimenter le menu déroulant `<select>`.  
`onchange="selectEmployee()"` va appeler, à chaque changement de valeur du select, la fonction javascript "selectEmployee" que nous allons voir:

```
function selectEmployee() {
    let employeeId = document.getElementById("employee_id").value;
    $.ajax({
        type : 'GET',
        url : '/apiFront/getEmployeeById',
        data : "id=" + employeeId,
    }).done(function(data) {
        $("#dataEmployee").html(data);
    });
}
```

Ici on emploie la librairie JQuery pour faire une requête AJAX (JQuery, étant déjà installé dans le projet, par soucis de temps j'ai utilisé l'outil Ajax).

La requête AJAX consiste à effectuer l'envoi et la réception des données d'un formulaire, sans rechargement de page. C'est utile notamment pour rendre l'expérience plus fluide. Dans notre cas on va s'en servir pour remplir notre formulaire, voyons comment cela fonctionne:

On envoie le formulaire à la route /apiFront/getEmployeeById qui correspond à la méthode getEmployeeById dans EmployeeController.

Le formulaire contient le numéro identifiant du salarié.

```
Route::prefix('apiFront')->name('apifront.')->middleware(['auth'])->group(function() {
    Route::get('/getEmployeeById', 'Api\EmployeeController@getEmployeeById');
});
```

Dans EmployeeController on a simplement la méthode getEmployeeById qui va s'occuper de renvoyer une vue remplie par les informations du salarié:

```
public function getEmployeeById(Request $request) {
    $auth = Auth::user(); //on a toujours la vérification de
    l'authentification, dans ce cas l'erreur
    if(empty($auth)){ // ne va pas bloquer le déroulement de la page,
    simplement le formulaire
        abort(404); // ne se remplira pas
    }

    $entreprise = $auth->entreprise;

    $idEmployee = $request->input('id');

    if(isset($idEmployee))
    {
        $employee = $entreprise->staff->where('id', $idEmployee)->first();

        if($employee){
            return view('api.employee.form.index',compact('employee'));
        }
    }
}
```

Comme déjà vu plus haut, on vérifie l'existence des données, et on les renvoie à une vue. Cette vue va s'incorporer dans le formulaire d'origine et l'alimenter avec ces nouvelles informations.

Quand on revient dans la vue du formulaire, on voit que les champs sont apparus, et le formulaire peut continuer jusqu'à sa validation

Ajouter un accident du travail

**Employé \***

JULES CESAR

**Employé**

JULES CESAR

**Genre**

Homme

**Numéro de sécurité sociale**

1200458753951

**Date de naissance**

04 / 08 / 2000

**Nationalité**

EEE

**Adresse**

ROUTE DE DOLE

**Code postal**

39800

**Adresse**

TOURMONT

**Date d'embauche**

27 / 04 / 2010

**Poste**

**Qualification professionnelle**

OP2

**Type de contrat**

CDI

**Ancienneté**

**Date \***

jj / mm / aaaa

**Heure \***

-- : --

**Type d'accident \***


Travail

**Lieu de l'accident \***

**Type du lieu de l'accident \***

Lieu de travail habituel

**N° SIRET du lieu de l'accident**



Si l'accident n'est pas survenu au sein de l'entreprise "ACT".

### Petite précision sur le CSRF (ou aussi XSRF)

Le CSRF ou *cross-site request forgery* est un type de vulnérabilité des services d'authentification.

Le but est, pour un utilisateur enregistré, de faire des requêtes d'un statut plus élevé, avec plus de droits (un administrateur par exemple).

Il faut connaître le lien permettant de faire l'action en question (par exemple un lien pour supprimer une personne sur un forum)

Envoyer un message ou transmettre un script (via une pseudo-image par exemple) contenant le lien en question, qui sera lancé avec les droits administrateur

Laravel, pour se prémunir de cette attaque, va générer un "token" (une suite de caractères aléatoires uniques) pour chaque session utilisateur active gérée par l'application. Il va être utilisé pour savoir si c'est bien la personne qui fait les requêtes dans l'application et puisque il est stocké dans la session et change à chaque génération de session, une tierce personne avec un autre token ne sera pas capable d'y accéder.

Donc, dans Laravel, à chaque formulaire POST, PUT, PATCH ou DELETE il faut soit inclure le token dans un champ input caché :

```
<input type="hidden" name="_token" value="{{ csrf_token() }}" />
```

Soit utiliser Blade avec @csrf qui ajoutera un champ caché avec le token pour la validation du formulaire.

Le middleware inclus par défaut dans le fichier de routes *web.php* va comparer son token de session au token envoyé par le formulaire.

Si le token du formulaire et le token de la session correspondent, on sait que la personne connectée est celle qui envoie le formulaire.

## Travail sur le remplissage automatique de pdf

La situation est la suivante : Corriger un formulaire administratif. On pouvait le télécharger mais il fallait le remplir "à la main".

Je me suis demandé s'il n'était pas possible de remplir le pdf automatiquement. J'ai donc demandé à mon tuteur si je pouvais y consacrer du temps.

Après recherche, j'ai trouvé une librairie pouvant remplir les PDF : [FPDF](#).

J'ai trouvé un script avec la bonne manière de faire, mais j'ai besoin qu'il supporte les cases à cocher. Dans la doc de FPDF je trouve un script qui fait le travail : [script93](#), et son évolution par un autre développeur pour ajouter les checkboxes : [FPDM](#).

Ce script est très utile car il va s'installer avec composer, se charger automatiquement, corriger les problèmes d'encodage et surtout, remplir mon PDF.

J'ai trouvé ma librairie, maintenant il me faut préparer mon pdf au remplissage. Il y a un paramètre à prendre en compte:

Est-ce que le document a des champs à remplir (ex. on clique dans le champ et on le remplit)?

- Si oui, on peut passer à l'étape suivante

- Si non, il faut créer les champs à remplir avec par exemple Adobe Acrobat (je n'ai trouvé aucun logiciel libre ou gratuit faisant ce travail).

Etape suivante : on va rendre le pdf compatible avec notre script avec [PDFtk](#) la version serveur est gratuite et nous suffira amplement.

Après installation : nous allons effectuer la commande :

```
pdftk pdforiginal.pdf output pdfmodifie.pdf
```

où pdfmodifie.pdf sera notre pdf rendu compatible.

On va faire la commande:

```
pdftk pdfmodifie.pdf generate_fdf output champs_a_remplir.fdf
```

(attention à ne pas avoir d'espace dans le chemin du fichier)

On va récupérer le nom des champs à remplir de notre pdf, nous nous en servons plus tard.

Maintenant que nous avons tout, on va l'installer (dans mon cas dans Laravel) dans notre environnement de travail.

Pour se faire, dans une application Laravel ou dans un autre projet avec Composer installé:

```
composer require tmw/fpdm
```

Les extensions ajoutées sur Laravel sont automatiquement chargées. Si ce n'est pas le cas pour vous il suffit d'ajouter dans le code :

```
require 'vendor/autoload.php'
```

Une fois que tout est installé, nous allons passer à l'implémentation dans le projet.

Dans mon cas, la vue et le controller étaient déjà présents, donc j'ai écrit dans le controller dédié aux arrêts de travail.

Il me manquait quelques champs dans la table personnels comme profession et nationalité, j'ai donc effectué la commande:

```
php artisan make:migration add_entreprises_column --table=entreprises
```

Dans le terminal, j'ai ensuite rempli le fichier

"database\migrations\2022\_02\_02\_134447\_add\_entreprises\_columns.php" ainsi créé.

On se retrouve avec deux méthodes. "Up", qui va s'activer avec la commande `hp artisan migrate`, et "down" qui va s'activer lors de la réinitialisation de la base de données, par exemple avec la commande `php artisan migrate:fresh --seed`.

```
class AddEntreprisesColumns extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::table('entreprises', function (Blueprint $table) {
            $table->string('numero_tel')->nullable();
            $table->string('email')->nullable();
            $table->string('numero_risque_SS')->nullable();
            $table->string('nom_service_sante')->nullable();
            $table->string('adresse_service_sante')->nullable();
            $table->string('CP_service_sante')->nullable();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        //
    }
}
```

En "up" on va simplement ajouter nos colonnes manquantes, profession et nationalité, en tant que chaînes de caractère, dans la table personnels.

En down on ne fait rien car le fichier qui crée la table s'occupe déjà de supprimer la table, mais on pourrait très bien demander à supprimer ces deux colonnes avec

```
$table->dropColumn(['numero_tel', 'email', 'numero_risque_SS', 'nom_service_sante', 'adresse_service_sante', 'CP_service_sante']);
```

Il me manque aussi des champs pour la table personnels, je fais donc la même démarche et ajoute:

```
Schema::table('personnels', function (Blueprint $table) {
    $table->string('profession');
    $table->string('nationalite');
});
```

Une fois les migrations faites, il ne faut surtout pas oublier de modifier les modèles. Dans les modèles on va spécifier ce qui va pouvoir être modifié. Pour cela on va dans le modèle "Entreprise" dans le dossier "app" :

```
class Entreprise extends Model
{
    use SoftDeletes;

    protected $fillable = [
        'reference',
        'nom',
        'ville',
        'adresse',
        'codepostal',
        'siret',
        'description'
    ];
}
```

En l'état des choses, on ne va pas pouvoir remplir d'autres champs que ceux-ci. Il faut donc rajouter nos colonnes à cette liste:

```
class Entreprise extends Model
{
    use SoftDeletes;

    protected $fillable = [
        'reference',
        'nom',
        'ville',
        'adresse',
        'codepostal',
        'siret',
        'description',
        'numero_tel',
        'email',
        'numero_risque_SS',
        'nom_service_sante',
    ];
}
```



```
'adresse_service_sante',
'CP_service_sante'
];
```

On fait la même chose pour le modèle Personnel

```
protected $fillable = [
'reference',
'sexe',
'nom',
'prenom',
'num_ss',
'date_naissance',
'adresse',
'cp',
'ville',
'qualification',
'statut',
'profession',
'nationalite',
'poste_id',
'date_entree',
'date_sortie',
'entreprise_id',
'email',
'telephone',
'commentaire'
];
```

Nous sommes maintenant prêts pour passer à l'étape suivante.

Pour commencer, on va créer une nouvelle méthode download, avec comme paramètre la requête POST et l'ID de l'accident, ainsi que les vérifications d'usage (utilisateur connecté, entreprise existante, etc.), en renvoyant vers une page 404 si l'un des paramètres est manquant:

```
public function download(Request $request, $accidentId)
{

    $auth = Auth::user();

    if (empty($auth)) {
        abort(404);
    }

    $entreprise = $auth->entreprise;
    if (empty($entreprise)) {
        abort(404);
    }

    $accident = $entreprise->accidents->where('id', $accidentId)->first();
    $personnel = $entreprise->staff->where('id', $accident->personnel_id)->first();

    if (empty($accident)) {
        abort(404);
    }
}
```

Dans notre fichier "champs\_a\_remplir.fdf" on va récupérer le nom des champs remplissables  
 Dans mon cas il est très long donc nous allons le tronquer:

```
%FDF-1.2
%âäÏÓ
1 0 obj
<</FDF
<</Fields
[
<</V O/T (heure constat accident)>>
<</V O/T (Code postal)>>
<</V O/T (date accident)>>
<</V O/T (date constat accident)>>
<</V O/T (Siège des lésions)>>
<</V / /T (témoin oui)>>
<</V / /T (autres victimes oui)>>
<</V O/T (date embauche)>>
<</V / /T (autres victimes non) >>
<</V O/T (date inscription accident)>>
<</V O/T (Code postal victime)>>
]
>>
>>
endobj
trailer
<</Root 1 0 R>>
%%EOF
```

On peut voir le nom des champs à remplir, et les checkboxes (qui sont des cases individuelles TRUE ou FALSE) Ce sont les noms que nous allons reprendre dans notre controller sous forme de tableau.

```
$fields = array(
//l'employeur
'accident travail' => $accidentTravail,
'accident trajet' => $accidentTrajet,
'Nom et prénom ou raison sociale de l'employeur' => $entreprise->nom,
'Adresse' => $entreprise->adresse, // numero & rue employeur
'Code postal' => $entreprise->codepostal, //CP employeur
'adresse 2 employeur' => $entreprise->ville, //commune, ville
'téléphone employeur' => $entreprise->numero_tel,
'Siret etablissement' => $entreprise->siret,
'Code RISQUE' => $entreprise->numero_risque_SS,
'Nom service santé' => $entreprise->nom_service_sante,
'adresse service santé' => $entreprise->adresse_service_sante,
'Code postal service santé' => $entreprise->CP_service_sante,
// //la victime
'N dimmatriculation' => $personnel->num_ss, // Num Secu victime
'clé immat' => $cleNumSS, // clé num secu victime
'sexe victime féminin' => $sexeVictimeF, // checkbox
'sexe victime masculin' => $sexeVictimeH, // checkbox
```

On peut voir le nom du champ en premier, associé à sa valeur (que l'on récupère de la base de donnée avec Laravel eloquent)

```
$accident = $entreprise->accidents->where('id', $accidentId)->first();
$personnel = $entreprise->staff->where('id', $accident->personnel_id)->first();
```

Pour les checkboxes, il faut faire les conditions manuellement. Dans ce cas avec des ternaires (On regarde si la valeur de type est égale à 'trajet'. Si oui : on donnera la valeur 'oui' à \$accidentTrajet qui commande la checkbox Trajet. Sinon on coche la case accident travail). On effectue ce genre de logique sur toutes les checkboxes:

```
//configuration des checkboxes
$accidentTravail = '';
$accidentTrajet = '';
($accident->type == 'trajet') ? $accidentTrajet = 'oui' : $accidentTravail = 'oui';

$sexeVictimeH = '';
$sexeVictimeF = '';
($personnel->sexe == 'Femme') ? $sexeVictimeF = 'oui' : $sexeVictimeH = 'oui';

$contratCDI = '';
$contratCDD = '';
$contratApprentiEleve = '';
$contratInterimaire = '';
$contratAutre = '';
if($personnel->statut == 'CDI'){ $contratCDI = "oui"; }
if($personnel->statut == 'CDD'){ $contratCDD = "oui"; }
if($personnel->statut == 'ApprentiEleve'){ $contratApprentiEleve = "oui"; }
if($personnel->statut == 'Interimaire'){ $contratInterimaire = "oui"; }
if($personnel->statut == 'Autre'){ $contratAutre = "oui"; }

}
```

J'ai cherché mais dans PHP 7.4, qui était la version utilisée avec Laravel 8.2, l'utilisation de Case n'est pas possible avec autre chose que des nombres entiers. Ce sera le cas à partir de PHP 8

Les dates nécessitent un peu de formatage pour coller avec le document pdf. On va les formater avec les fonctions natives de php `date_create` et `date_format` qui créent un objet date avec la classe `DateTime` et le formattent <https://www.php.net/manual/fr/datetime.format.php>:

date en entrée : {**2022-03-17**}

```
$date = date_create($personnel->date_naissance);
$dateNaissance = date_format($date, 'dmY');
```

date en sortie : {**17032022**}

On va calculer l'ancienneté du personnel en comparant la date d'entrée du salarié avec la date d'aujourd'hui avec la classe `Carbon`, qui hérite de la classe `DateTime` utilisée plus haut (notons bien qu'il est toujours avisé de déclarer les classes dans le namespace pour éviter de l'appeler par son chemin à chaque fois et pour plus de lisibilité)

```
$anciennete = Carbon::parse(Carbon::now())
->diffInDays($personnel->date_entree);
```

Ici on va parse (transformer en chaîne de caractères) la différence en jours entre la date d'aujourd'hui et la date d'entrée.

Enfin, on va créer un nouveau fichier .pdf, qui prendra les valeurs et le nom qu'on lui aura donné:

```
$pdfname = 'AT_'.$personnel->nom.'_'.$personnel->prenom.'_'.$dateAccident_store;
$pdf = new \FPDF('cerfa/cerfa_AT.pdf');
$pdf->useCheckboxParser = true;
$pdf->Load($fields, true);
$pdf->Merge();
$pdf->Output('I', $pdfname);
```

- On donne le nom du pdf
- On crée un nouvel objet FPDF avec comme base notre formulaire vide
- On spécifie que l'on veut utiliser les checkboxes
- On va charger le tableau avec les valeurs remplies
- On fusionne les valeurs remplies avec le pdf vide
- On sort le pdf rempli avec le nom donné plus haut (<https://www.fpdf.org/en/doc/output.htm>) avec l'option "I" qui signifie qu'on va envoyer directement le pdf au navigateur qui va l'afficher, et si il ne peut pas, le télécharger.

## 7 - Présentation du jeu d'essai

On commence par créer un salarié:

Ajouter un salarié

**Genre \***

Homme

**Nom \***

Chirac

**Prénom \***

Jacques

**Numéro de Sécurité Sociale et Clé \***

132115934809

**Date de naissance \***

29 / 11 / 1932

**Adresse \***

123 rue ancienne de l'Elysée

**Code postal \***

75008

**Ville \***

Paris

On remplit le formulaire d'accident du travail:

## Accidents du travail

[Retour](#)

Ajouter un accident du travail

**Employé \***

CHIRAC Jacques

**Employé**

CHIRAC Jacques

**Genre**

Homme

**Numéro de sécurité sociale**

132115934809

**Date de naissance**

29 / 11 / 1932

**Nationalité**

EEE

**Adresse**

123 rue ancienne de l'Elysée

**Code postal**

75008

**Adresse**

Paris

**Date d'embauche**

04 / 03 / 1997

**Poste**

**Qualification professionnelle**

BAC+2

**Type de contrat**

Interimaire

**Ancienneté**

**Date \***

15 / 03 / 2022

**Heure \***

22:22

**Type d'accident \***

Travail

**Lieu de l'accident \***

Entreprise

**Type du lieu de l'accident \***

Au cours du trajet entre le travail € ▾

**N° SIRET du lieu de l'accident**

Si l'accident n'est pas survenu au sein de l'entreprise "ACT".

**Activité de l'employé lors de l'accident \***

Pause déjeuner

Limité à 255 caractères maximum.

**Nature de l'accident \***

L'employé ramenait son café et a loupé une marche.

Limité à 255 caractères maximum.

On sélectionne l'accident que l'on veut télécharger ou imprimer:

Liste des accidents du travail de l'entreprise : **ACT**

N°	Date Accident	Salarié	Actions
15	15/03/2022	<b>CHIRAC</b> Jacques Numéro de Sécurité Sociale : 132115934809	Télécharger Modifier Supprimer
13	03/03/2022	<b>JULES</b> CESAR Numéro de Sécurité Sociale : 1200458753951	Télécharger Modifier Supprimer
12	16/02/2022	<b>MORILLE-ROY</b> Lucas Numéro de Sécurité Sociale : 189025935049706	Télécharger Modifier Supprimer

On vérifie les informations entrées et on enregistre ou imprime le pdf.

**cerfa**  
N° 14483\*02  
DAT-PRE

### DÉCLARATION D'ACCIDENT DU TRAVAIL ☒ D'ACCIDENT DE TRAJET ☐

(Articles L. 461-1 à L. 461-4 et articles R. 461-2, R. 461-3, R. 461-4 et R. 461-11 du Code de la sécurité sociale)  
L'employeur verse à la caisse primaire d'allocations familiales de sa commune, les trois premiers jours de la déclaration, par lettre recommandée  
avec accusé de réception, les plus tards 15 jours après la survenue de l'accident et envoie la déclaration à la caisse d'allocations familiales.

**L'EMPLOYEUR** (établissement d'affiliation permanent de la victime) (se reporter à la notice)

Nom et prénom ou raison sociale de l'employeur : **ACT**

Adresse : 1 Impasse de la source  
3 9 0 0 0 MILLERVAINE N° de Téléphone : + 3 8 5 5 1 4 5 5 7

Cette page : N° SIRET de l'établissement d'origine : 8 0 1 5 3 7 4 0 2 0 0 0 1 5 N° de Rigue Sécurité Sociale : 1 2 3 4 5

Nom du service de travail : 123 Adresse : 123 Cette page : 1 2 3 4 5

**LA VICTIME** (se reporter à la notice)

N° d'immatriculation : 1 3 2 1 5 9 3 4 8 0 9 A défaut, sexe ☒ M ☐ F Date de naissance : 2 9 1 1 1 9 3 2

Nom et prénom : **CHIRAC, JACQUES**

Adresse : 122 rue au château de la Vierge Nationalité : Française ☐ CEE, Suisse ☒ Autre ☐

Cette page : Date d'embauche : 0 4 0 3 1 9 9 7 Profession : Manœuvre Qualification professionnelle : R47 Ancienneté dans le poste de travail : 0136 Jours

Contrat de travail : CDI ☐ CDD ☐ Apprenti/élève ☐ Intérimaire ☒ Autre ☐

**LES INFORMATIONS RELATIVES À L'ACCIDENT** (se reporter à la notice)

Date : 1 5 0 3 2 0 2 2 Heure : 1 4 4 3

Lieu de l'accident : Entreprise

Précisions complémentaires sur le lieu de l'accident et sur le temps : ☐ au cours du trajet entre le domicile et le lieu de travail ☐ au cours du trajet entre le travail et le lieu de travail ☐ au cours d'un déplacement pour l'employeur

Nature de l'accident : Pause déjeuner

Objet dont le contact a blessé la victime : Le café brûlant

Éventuelles réserves motivées (joignez, si besoin, une lettre d'accompagnement) : La température exacte du café reste à déterminer

Siège des lésions : Main droite

Nature des lésions : Blessures légères

La victime a-t-elle été transportée à : Finimment L'accident a-t-il fait d'autre(s) victime(s) ? OUI ☒ NON ☐

Horaires de travail de la victime le jour de l'accident : de 0 8 0 0 à 1 2 0 0 et de 1 4 0 0 à 1 8 0 0

Accident constaté : ☒ par l'employeur ☐ par ses préposés ☐ déclaré par la victime

Conséquences : SANS ARRÊT DE TRAVAIL ☒ AVEC ARRÊT DE TRAVAIL (\*) ☐ Décès ☐

Un rapport de police a-t-il été établi ? NON ☒ OUI ☐ par qui ?

**LE TÉMOIN ou LA PREMIÈRE PERSONNE AVISÉE** (se reporter à la notice correspondante)

Le témoin ☐ ou la 1ère personne avisée ☒ (en cas d'absence de témoin)

Nom et prénom : Jean michaud Adresse : 122 rue de la Vierge, Ville Cette page : 1 2 3 4 5

**LE TIERS**

L'accident a-t-il causé par un tiers ? OUI ☐ NON ☒

Si OUI, nom et adresse du tiers : Société d'assurance du tiers

Nom et prénom du signataire : Signature

Fait à : le :

(\*) : Important, si l'accident a entraîné un arrêt, remplissez immédiatement l'attestation de salaire S 6202.

DAT-PRE 86200h

## 8 - Description de la veille

Suivi de quelques comptes liés aux technologies employées:

### - LinkedIn

- ANSSI - Agence nationale de la sécurité des systèmes d'information ANSSI - Agence nationale de la sécurité des systèmes d'information - 197 998 abonnés
- UI/UX User Experience Interactive Designer / Wireframes - 447 248 membres
- UI Designer and UI Developer - 201 675 membres
- Designers Talk : Graphic, web, digital design and creative professionals group - 210 304 membres
- JavaScript - 272 384 membres
- Front End Developer Group - 355 533 membres
- Stack Overflow - 877 574 abonnés
- CNIL - 155 997 abonnés

### - Twitter

- php.net
- Povilas Korop
- Laravelphp
- Codrops
- Lior Chamla
- Nuno Maduro
- La quareature du Net
- Aurélie Vache
- Sara Soueidan
- Beyond Code

### - Youtube

- Le designer du web
- Nord Coders
- Lior Chamla
- Micode
- Underscore
- Basti UI
- Développeur Libre
- Korben
- Le frère Codeur

### - Podcasts

- Good Morning Web
- Artisan Développeur
- L'Octet vert
- Tech café
- Webausaures



## - Sites

- Grafikart
- CSS-tricks
- la ferme du web
- le site de l'ANSSI

## 9 - Description d'une situation de travail ayant nécessité une recherche

Durant l'exercice de mon stage, j'ai eu à chercher de la documentation sur l'API d'une application nommé PLESK, C'est un logiciel de gestion de serveurs et de domaines. L'objectif était de faire une fiche récapitulative des commandes de l'api. La documentation est en anglais, dont voici un extrait traduit.

### About REST API

You interact with Plesk via REST API by sending API requests in the form of HTTP requests. REST API is based on OpenAPI Specification 2.0 (formerly Swagger specification).

Only the Plesk administrator can use REST API.

### A propos de l'API REST

Vous interagissez avec Plesk via l'api REST en envoyant les requêtes sous la forme de requêtes HTTP. L'api est basée sur les spécifications OpenAPI 2.0 (anciennement les spécifications Swagger)

There are various tools for working with API requests:

API clients, software with a graphical interface for sending and receiving API requests, for example Postman.

Command line tools such as curl.

Swagger tools.

Il y a plusieurs outils pour faire les requêtes API:

Des clients API, des logiciels avec une interface graphique pour envoyer et recevoir des requêtes api, Postman par exemple.

Des outils en ligne de commande comme curl.

Des outils Swagger.

Regardless of the tool you choose, a typical API request consists of the following components:

HTTP method

URL

Data type

Transferred data

Authorization methods

Indépendamment de l'outil que vous choisissez, une requête API typique contient les composants suivants:

- La méthode HTTP
- L'URL
- Le type de données
- Les données transférées
- Les méthodes d'accès

Let's look closely at each one of them.

#### HTTP method

The data specified in the request determines what response you get back. Usually, the response contains a body with the information you have requested and an HTTP status code. The HTTP status code indicates if the request was successful or not. In general, if you receive a 2xx status code, the request was successful. Both 4xx and 5xx status codes mean that there has been an error processing your request. 4xx status codes tell you that the error is on your side (for example, bad syntax of the request), while 5xx status codes mean that the error is on the server's side.

To choose which HTTP method to use, see the REST API autogenerated reference. Log in to Plesk, go to Tools & Settings > Remote API (REST) (under "Server Management"), and then click API Reference and Playground (you will need to provide the server root or administrator user credentials).

Les données spécifiées dans la requête déterminent quelle réponse vous aurez. Habituellement, la réponse contient un corps avec l'information que vous avez demandé et un code de statut HTTP.

Le code HTTP indique si la requête a été fructueuse ou non.

En général, si vous recevez un statut 2xx, la requête a été réussie. Les codes 4xx et 5xx signifient qu'il y a eu une erreur dans le traitement de la requête.

Un code 4xx dira que l'erreur est de votre côté (par exemple, une mauvaise syntaxe dans la requête), tandis que les codes 5xx signifient que l'erreur est du côté serveur.

Pour choisir quelle méthode HTTP utiliser, allez voir la référence autogénérée de l'API. Connectez vous sur Plesk, allez dans outils et réglages > API Distant (REST) (dans "gestion de serveur"), ensuite cliquez sur Terrain de jeu et Liste de référence de l'API (vous devrez fournir les identifiants serveur ou administrateur)

source : [Plesk Api Documentation](#)

---

## 10 - Conclusion

---

Ce stage, et plus généralement cette formation a été pour moi un enrichissement sans pareil. En neuf mois j'ai découvert et pratiqué une profession qui me paraissait inatteignable il y a quelques années.

Durant ce stage, j'ai pu consolider mais surtout éprouver mes nouvelles connaissances au milieu du travail, dans un contexte de rentabilité et dans les contraintes de temps. Plus que les technologies employées, c'est surtout cela qui a été formateur.

J'ai appris que je pouvais m'adapter à toutes sortes de situations grâce à l'appui de mes pairs.

J'ai fait toutes les erreurs possibles, j'ai appris de chacune, et cela ne me gêne pas d'en faire d'autres car une erreur peut nous apprendre autant que des dizaines d'heures de cours.

L'horizon est large et je ne sais pas encore où je serais dans quelques années, mais une chose est sûre, c'est que je continuerai le développement web.

### Remerciements :

**Marian Denys**, gérant de l'entreprise DMWeb qui m'a fait confiance et m'a accueilli durant ce stage, m'a confié des tâches variées et enrichissantes.

**Igor**, développeur chez DMWeb, qui a eu la patience de me corriger et m'a appris beaucoup.

**Xavier**, alternant Online Formapro chez DMWeb qui m'a accompagné dans mes déboires et m'a soutenu.

Mon formateur **Alain Merucci**, que je remercie de son abnégation à enseigner un métier si vaste et riche en connaissances à des néophytes. Nous ne pensions pas qu'autant de connaissances rentreraient en aussi peu de temps, et c'est grâce à lui.

Mes camarades de promotion qui ont créé une synergie d'apprentissage, une ambiance bienveillante, chacun m'a appris à sa manière.

Remerciement spécial à **Vanessa Maquet**, assistante administrative à Online Formapro qui a fait tant pour toutes les personnes de la formation.