

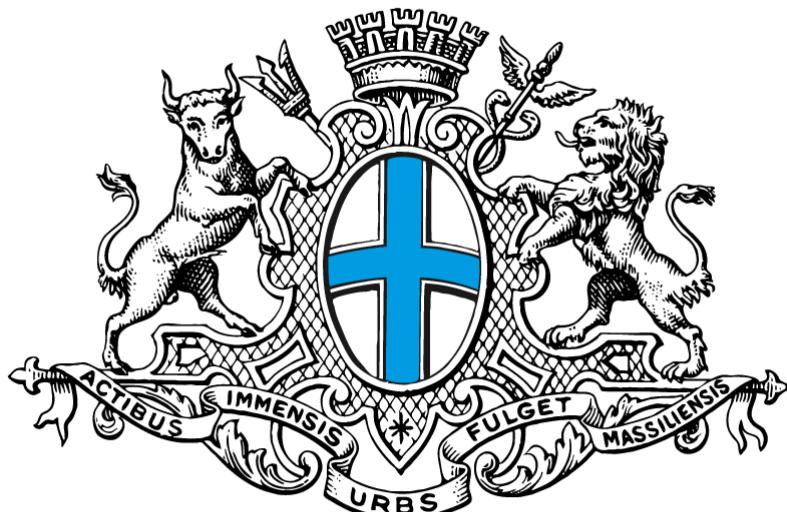
LEAL DIAS SEMEDO SARA

BTS Services Informatiques aux Organisations
Option SLAM

Lycée Technique Marie Curie 16
Boulevard Jeanne d'Arc 13005 Marseille

Année scolaire 2023 / 2024

VILLE DE



MARSEILLE

Stage de 1^o année de BTS SIO
Du 20 mai 2024 au 28 juin 2024

REMERCIEMENTS

Durant ma première année de BTS SIO, j'ai effectué un stage dans l'entreprise de la division développement logiciel de la ville de Marseille.

Je tiens à remercier Madame Sidoli Valérie chargé de la gestion en ressources humaines pour avoir activement aidé ma candidature et le suivi de la faisabilité de mon stage.

Monsieur Henry Kevin, mon tuteur de stage, qui a su me mettre à l'aise et faire preuve d'écoute lorsque j'en avais besoin, ainsi que son appui dynamique au cours de mon stage.

Belin Julien, Rocheron Matthieu et Lautard Mathieu. Ils n'étaient pas mes tuteurs de stage par écrit, mais ils étaient toujours là quand nous avions besoin d'eux et toujours souriants, avec plus de projets et de connaissances à nous faire découvrir. Qu'il s'agisse de l'apprentissage de Laravel, de l'immersion dans le monde professionnel, ou de l'apprentissage de techniques de progrès personnel.

Et l'équipe pédagogique du Lycée Marie-Curie pour son soutien et ses enseignements éducatifs et techniques.

Sommaire :

I) Contexte du stage	1
1) Présentation de l'entreprise	
2) Maintien du Système d'Information de la Ville de Marseille	
3) Présentation du matériel et du logiciel utilisés	
II) Travail réalisé	5
1) Présentation du travail	
2) Planning de travail	
3) Tâches réalisées	
4) Compétences professionnelles mises en œuvre	
III) Conclusion	18
1) Bilan sur le projet	
2) Bilan personnel	
3) références	

I) Contexte du stage

Dans le cadre de mon stage de première année en BTS services informatiques aux organisations option B, solutions logicielles et applications métiers (SIO SLAM) au lycée Marie-Curie, j'ai effectué un stage au sein de la ville de Marseille. Du 21 mai au 28 juin 2024, j'ai effectué un stage au sein de la division développement logiciel de la ville de Marseille. Au cours de ce stage, j'ai pu réaliser plusieurs tâches à la demande des utilisateurs et l'immersion totale dans une organisation.

Dans ce rapport, je présenterai dans un premier temps la collectivité territoriale de la ville de Marseille et les différentes cyber sécurités apportées par la ville de Marseille. Dans un second temps, les matériels et les logiciels utilisés au sein de la division développement logiciel. Dans un troisième temps, le travail réalisé au sein de la ville de Marseille et enfin je vais conclure mon rapport sur le bilan du projet et mon bilan personnel.

1. Présentation de L'entreprise

Les collectivités territoriales sont des personnes morales de droit public, tout comme l'État et les établissements publics, dotées de compétences propres et administrées de manière décentralisée par des organes élus au suffrage universel direct. Elles ont pour mission de prendre en charge les intérêts de la population dans un territoire spécifique.

Comme toutes les collectivités territoriales, la Ville de Marseille est gouvernée par un conseil municipal comme organe délibérant, et par un maire comme organe exécutif. L'organigramme de la ville de Marseille, illustre la structure descendante de la ville et puis pour ses 9 directions générales des services, l'organigramme est présenté sous forme de trèfle afin de ne pas différencier les différentes directions. L'organigramme en trèfle est un type d'organigramme qui convient parfaitement aux organisations qui ont un mode de gestion collégiale. Autrement dit, l'organigramme en trèfle est adapté aux organismes qui mettent leurs différents postes ou fonctions aux mêmes pieds d'égalité.

Les collectivités territoriales disposent d'un budget autonome et gèrent leurs ressources (impôts locaux, dotations de l'État) dans le cadre défini par la loi. Elles possèdent également un pouvoir réglementaire significatif.

La décentralisation implique le transfert de compétences de l'État vers les collectivités territoriales (régions, départements, communes), leur conférant une autonomie juridique et budgétaire ainsi qu'un pouvoir de décision dans la gestion quotidienne selon les règles établies par l'État.

L'article 72-3 de la Constitution garantit le principe de libre administration, essentiel à la décentralisation. Les conseils des collectivités territoriales sont élus au suffrage universel direct, et il incombe au législateur de définir les compétences respectives de l'État et des collectivités.

L'administration décentralisée gère les affaires locales de manière autonome, avec un contrôle de l'État a posteriori. Quant à l'administration déconcentrée, elle représente les intérêts de l'État au niveau local.

Le territoire est central dans l'identification des collectivités territoriales. Le législateur définit les principes fondamentaux de la libre administration pour protéger ces entités contre d'éventuelles ingérences du pouvoir exécutif, un principe affirmé à plusieurs reprises par le Conseil constitutionnel.

La Ville de Marseille utilise deux modes de gestion : la régie/gestion directe, où elle assure directement le fonctionnement des services publics avec ses propres moyens et agents, et la gestion indirecte ou déléguée, où elle confie la gestion à une entité publique ou privée tout en conservant un contrôle sur la conformité à l'intérêt général.

je faisais partie de la DGA TRANSFO. Direction Générale Adjointe. Transformer nos pratiques. Dans le service de la fabrique des solutions métiers, dans la division développement logiciel.

En conclusion, travailler à la Ville de Marseille signifie être partie prenante d'une équipe de 17 000 agents engagés dans la transformation durable de la cité, en respectant les principes d'égalité, de mutabilité, d'équité, d'adaptabilité et de continuité pour répondre aux besoins d'intérêt général de manière efficace et équitable.

2) Maintien du Système d'Information de la Ville de Marseille

La ville de Marseille a été confrontée à diverses attaques informatiques et à des pannes de serveurs par le passé, ce qui a mis en lumière la nécessité de renforcer la sécurité et la réactivité de son système d'information. Pour pallier ces problèmes, plusieurs mesures ont été mises en œuvre :

Les serveurs de la ville sont stockés dans plusieurs datacenters afin de garantir la sauvegarde des données. L'un des problèmes récurrents étant la surchauffe, une attention particulière est portée à la climatisation des salles serveurs.

Pour assurer une continuité de service et éviter la perte de données, la ville de Marseille utilise une stratégie de stockage répartie. Les données critiques sont dupliquées sur plusieurs serveurs, garantissant ainsi une haute disponibilité et une récupération rapide en cas de défaillance.

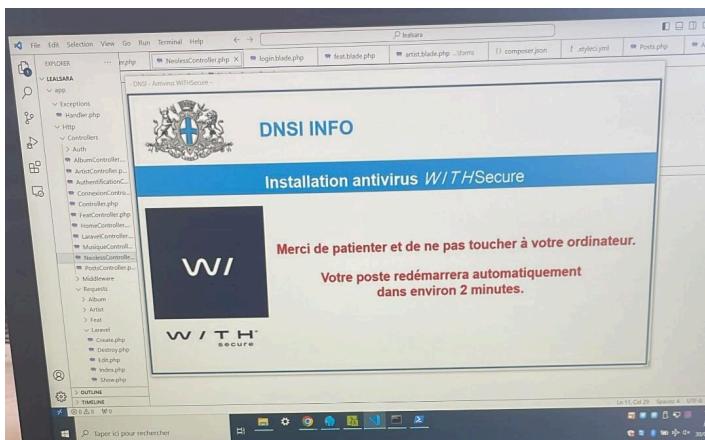
La sensibilisation des acteurs est cruciale pour renforcer la sécurité du système d'information. La ville de Marseille a mis en place des campagnes de phishing éthique. Cette pratique consiste à envoyer des emails de phishing simulés aux employés pour évaluer leur réaction. Les employés qui cliquent sur les liens reçoivent ensuite une sensibilisation détaillée sur les bonnes pratiques à adopter et les erreurs à éviter, améliorant ainsi la vigilance générale contre les cyberattaques.

Un autre point critique de la sécurité est la protection contre les injections SQL, une technique courante utilisée par les attaquants pour exploiter les vulnérabilités des applications web. Les injections SQL permettent à un attaquant d'exécuter des requêtes arbitraires sur la base de données, potentiellement compromettant des informations sensibles. Pour contrer ce type d'attaque, la ville de Marseille s'assure que les développeurs adoptent des pratiques de codage sécurisées, telles que l'utilisation de requêtes paramétrées et de procédures stockées.

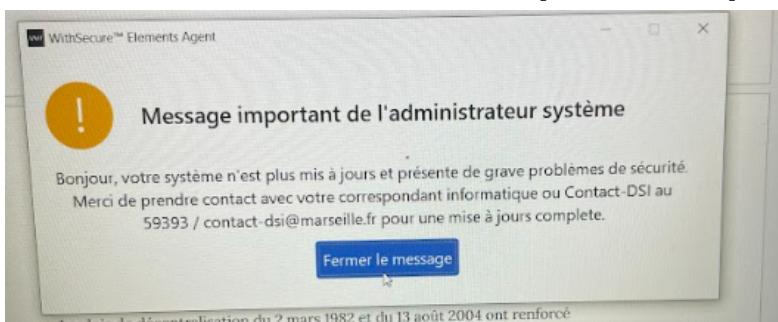
La maintenance des systèmes inclut également la mise à jour régulière des langages de programmation et des infrastructures logicielles utilisés. Les versions obsolètes peuvent contenir des failles de sécurité que les développeurs ne corrigent plus. En mettant à jour régulièrement ces composants, la ville minimise les risques associés aux vulnérabilités connues et bénéficie des améliorations de sécurité apportées par les nouvelles versions.

Les ordinateurs sont régulièrement mis à niveau pour une sécurité optimale grâce à l'entreprise Withsecure, qui effectue fréquemment des scans pour détecter les vulnérabilités du réseau et des actifs. Cette entreprise offre des capacités accrues de détection et de sécurité contre les cyberattaques et les violations de données. Elle assure également une protection des endpoints via le cloud pour bloquer les menaces avancées, automatisées et ciblées.

Voici un exemple d'installation d'un antivirus sur un ordinateur à l'aide de Withsecure.



Voici un exemple de messages que j'ai reçus à plusieurs reprises lorsqu'une mise à jour n'avait pas été effectuée. Il me demandait de contacter au plus vite un responsable de la division.



En conclusion, de nombreuses mesures ont été mises en place pour améliorer la sécurité et la réactivité du système d'information de la ville de Marseille, mais ce que je viens d'évoquer n'est qu'une petite partie de leur sécurité. La sécurité informatique est un domaine vaste et complexe. Un mois de stage n'est pas suffisant pour appréhender toutes les actions menées par la ville, mais il est clair que la ville s'engage activement dans la protection de son infrastructure et de ses données.

3) Présentation du matériel et des logiciels utilisés.

Au cours de mon stage, j'ai eu l'occasion d'utiliser une large gamme de matériel et de logiciels sur le marché, qu'ils soient commerciaux, de type "front end" ou "back end".



Tout d'abord nous avons utilisé du HTML qui signifie « HyperText Markup Language » qu'on peut traduire par « langage de balises pour l'hypertexte ». Il est utilisé afin de créer et de représenter le contenu d'une page web et sa structure. L'hypertexte désigne les liens qui relient les pages web entre elles, que ce soit au sein d'un même site web ou entre différents sites web. Les liens sont un aspect fondamental du Web.



Pour continuer, je vais vous présenter l'un des langages essentiels au langage html, le CSS (pour Cascading Style Sheets), soit feuilles de style en cascade, est un langage de feuille de style utilisé pour décrire la présentation d'un document écrit en HTML. Le CSS décrit la façon dont les éléments doivent être affichés à l'écran, sur papier, à l'oral ou sur d'autres médias. Le CSS est l'un des langages principaux du Web ouvert.



Ensuite, je vais introduire le Javascript. Le Javascript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. À chaque fois qu'une page web fait plus que simplement afficher du contenu statique (affiche du contenu mis à jour à des temps déterminés, des cartes interactives, des animations 2D/3D, des menus vidéo défilants, ou autre), JavaScript a de bonnes chances d'être impliqué. C'est la troisième couche des technologies standards du web, les deux premières étant le html et le css. Nous avons parfois eu besoin d'une bibliothèque javascript, la bibliothèque jQuery.

 Une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web.



De plus, nous avons utilisé le langage de programmation PHP (Hypertext Preprocessor) qui est un langage de scripts généraliste et Open Source, pour gérer des pages web dynamiquement via généralement un serveur HTTP. Le PHP est un langage interprété, cela veut dire que celui-ci doit être interprété par un serveur afin d'afficher correctement le contenu demandé. Il est bien évidemment possible d'utiliser php en local, comme un des serveur local que l'on à utiliser, MySQL.



Nous avons ensuite utilisé phpmyadmin, une application Web de gestion des systèmes de gestion de bases de données MySQL et MariaDB, écrite principalement en PHP et distribuée sous la licence GNU GPL. Grâce à

 ,un environnement de développement universel portable, isolé, rapide et puissant pour

Windows. Il est conçu pour simplifier la configuration et la gestion des environnements de développement web locaux, et il est livré avec un ensemble d'outils et de services qui sont couramment utilisés dans le



développement web, tels que PHP et MySQL. Mais aussi Composer un gestionnaire de dépendances PHP qu'on a pu utiliser pour la cybersécurité du site web.

Il possède sa propre architecture de services qui gère les services de manière à la fois synchronisée et non bloquante.



Laravel est un framework de construction d'applications web modernes et complètes. Laravel combine les meilleurs packages de l'écosystème PHP pour offrir le framework le plus robuste et le plus convivial pour les développeurs. Laravel offre des fonctionnalités puissantes telles que l'injection de dépendances, une couche d'abstraction de base de données expressive, des files d'attente et des tâches planifiées, des tests unitaires et d'intégration, et bien plus encore.



Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur le bureau et est disponible pour Windows, macOS et Linux. Il est livré avec un support intégré pour JavaScript, TypeScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages et runtimes.



Lunacy est un logiciel de conception graphique vectorielle de nouvelle génération pour l'UI/UX et la conception web. Il combine les meilleures fonctions des apps de design pour offrir l'expérience la plus agréable. Lunacy comprend une bibliothèque graphique intégrée, des outils de l'IA et permet de collaborer sur des projets en temps réel.



Trello est un logiciel de gestion de tâches. Il propose une série de fonctionnalités qui permettent d'ordonner d'une façon visuelle et logique les tâches à faire ainsi que les notes ou les idées qui puissent surgir. C'est aussi le même principe qu'un cloud, une fois avoir indiqué les codes d'accès, les documents sont accessibles depuis n'importe quel terminal.



Pendant mon stage, j'ai réalisé tous mes projets sur un ordinateur portable 11 generation intel core i5, sous windows 10 pro. L'ordinateur était protégé par un câble en cas de vol. J'avais une connexion ethernet pour le wifi. L'ordinateur est constamment mis à jour, et est très bien protégé administrativement. Avec un mot de

mot de passe recommandé pour un mot de passe optimal. Une phrase composée, avec des chiffres et des lettres spéciales. Ces mots de passe sont changés régulièrement comme recommandé.

II) Travail réalisé

Pendant mon stage, j'ai eu la chance de travailler sur plusieurs projets avec une équipe très compétente et sympathique.

Ce service est le seul à faire du développement à proprement parler, les autres services développant avec des prestataires de services.

La demande est à l'origine de tout, puis le comité de la demande se réunit pour décider de la faisabilité de la demande. Une fois la demande étudiée, ils décident si elle doit être développée en interne ou par un prestataire de services.

1) Présentation du travail.

J'ai fait deux gros projets pour la ville de Marseille. un projet personnel pour se mettre dans le bain et comprendre les outils que nous allions utiliser, et puis nous avons pu faire un projet pour la ville de Marseille sur la météo. Belin julien a été notre demandeur, il nous a demandé plusieurs choses qu'il souhaitait dans le site. Tout d'abord, il a exprimé son souhait que la ville de marseille ait toujours voulu un site météo bouche-du-rhône, afin d'avoir des alertes en fonction de la situation, et de permettre aux développeurs de récupérer toutes les données météo en fonction du jour et du temps (api), ainsi qu'une iframe météo qui pourrait être ajoutée à n'importe quelle page afin d'afficher la météo sur leur site. A côté de cela, une page de documentation de l'api, car il est important de comprendre son utilité et les fonctionnalités de l'api que nous avons construites. Pour le côté cybersécurité, j'ai pu lancer une API REST avec deux pages identifications, et lorsque l'utilisateur s'inscrit, que l'on puisse lui attribuer un token pour que l'utilisateur ne puisse faire des demandes pouvant altérer notre site.

2) Planning de travail

Pour notre premier projet, nous avons pris une semaine pour le réaliser. Il n'y avait pas vraiment de planning, il s'agissait plutôt de ce que nous pouvions faire, à notre rythme, tout en comprenant les bases de Laravel, et les différents langages de programmation PHP, HTML, CSS, et JAVASCRIPT. C'était la phase d'apprentissage.

Lors du projet pour la ville de Marseille, en revanche, nous nous sommes mis en immersion totale dans l'organisation. Pour le projet, nous avons utilisé la méthode sprint agile, qui nous a permis d'optimiser notre productivité et de mieux nous adapter aux changements grâce à des cycles de développement courts et agiles.

Tout d'abord, pour l'immersion, nous avons mis des post-it sur un tableau avec les différentes étapes de la construction du site web.

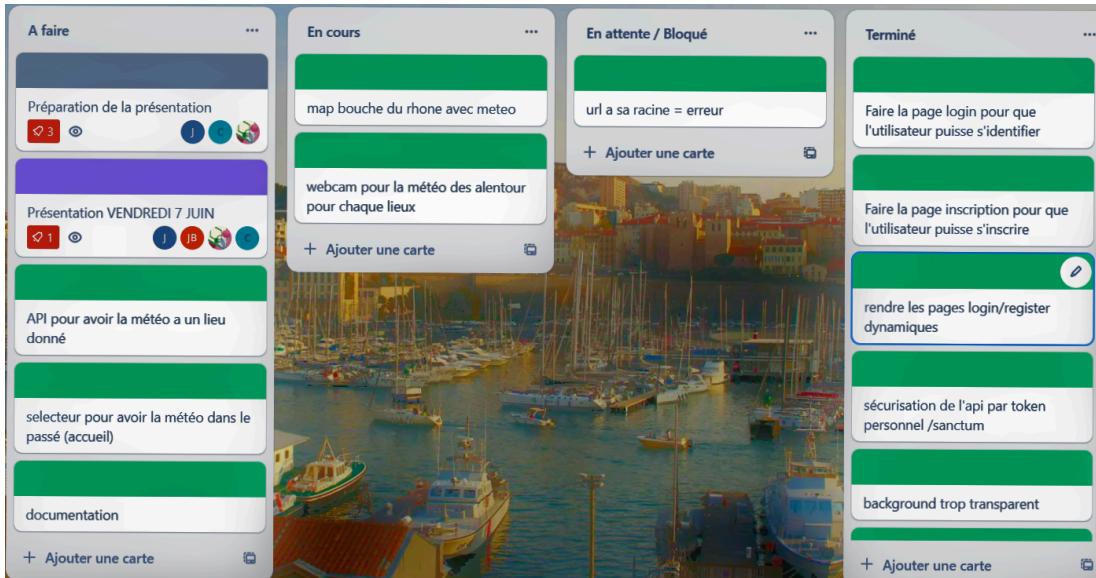
Pour nous organiser, nous nous sommes tournés vers trello, le logiciel de gestion des tâches. Mais avant cela, nous avons essayé d'utiliser le logiciel de gestion des tâches que la ville utilise, car il n'est pas conseillé de lier des données à des sites produits par un autre pays. En effet, si le pays décide de couper ou d'obtenir

l'information, la ville perdra toutes les données qu'elle a soigneusement introduites dans les logiciels. Cela signifie que la ville devra dépendre de ces sites et qu'elle sera laissée pour compte si le gouvernement du pays décide de tout couper.

Nous nous sommes imaginées ce que nous pourrions faire en une semaine, et avons rajouté une étiquette en violet.

7 JUIN

Pour continuer, nous avons ajouté les post-it que nous avions écrits sur le tableau numérique de Trello, en répartissant les différentes tâches.



Notre objectif étant d'en faire assez pour organiser une réunion à la fin de la semaine afin de présenter nos progrès et de voir si le client est satisfait, et pouvoir émettre son avis sur notre production.

3) Tâches réalisées

J'étais chargé de la documentation et de la sécurité de l'api. C'est de cela que je vais parler.

Tout d'abord, j'ai fait une maquette sur Lunacy pour faciliter la projection de la façon dont j'allais faire la page "documentation de l'API":

Pour continuer, j'ai commencé à faire le html et le css de la page en attendant le socle Laravel pour pouvoir l'intégrer dedans. j'ai fait un dossier avec dedans le html et le css ainsi qu'un dossier pour les images que je dois mettre dans le html.

	img	03/06/2024 16:01	Dossier de fichiers
	documentation_api.css	03/06/2024 15:54	Fichier source CSS 5 Ko
	documentation_api.html	03/06/2024 16:02	Fichier source HTML 3 Ko

Vous vous demandez sûrement ce qu'est le check vert en bas à gauche des documents.



C'est TortoiseSVN, un logiciel gratuit pour permettre de collaborer en groupe en obtenant des informations sur le statut des différents fichiers que nous avions changés, ajoutés, supprimés ou renommés, et quels fichiers ont été changés et livrés par les autres.

Il est en vert, ce qui signifie que tous les fichiers sont à jour.

Au cours du projet, j'ai dû adapter mon code html à php, je vais donc montrer les principaux changements que l'on peut rencontrer, et aussi me permettre de montrer le codage que j'ai pu faire.

Ces trois codes, sont pour la même chose, lier ma page css pour permettre la modification de ma page d'un point de vue plus esthétique.

Tout d'abord, le plus basique, la manipulation pour lier ma page à un .html. Il suffit d'ajouter un lien avec un nom et de trouver son répertoire.

```
<link rel="stylesheet" href="C:\laragon\www\meteomarseille\documentation_api\documentation_api.css">
```

Un peu la même règle, mais d'abord nous ajoutons le répertoire avec la classe asset, qui facilite la collecte, le regroupement et l'affichage des assets (js, css, img) car elle ne nécessite pas de répertoire.

```
<link href="{{ asset('css/Api.css') }}" rel="stylesheet">
```

```
@extends('laracrud.layouts.app')
@section('breadcrumb')

@endsection
@section('header')


# Home page projet


@endSection
@section('tools')

@endSection
@section('content')
<section>
```

La dernière méthode est d'ajouter un extends, qui permet de faire une structure définie de sections et de l'ajouter à la page choisie, qu'on a définie comme ".blade.php" le moteur de modèles par défaut du framework Laravel. Il permet d'utiliser des variables, des boucles, des instructions conditionnelles et d'autres fonctionnalités de PHP directement dans le code HTML.

Les différentes sections ont été ajoutées pour éviter la reproduction des mêmes lignes de code, comme le menu dans la section header.

Au lieu d'écrire ce script sur chaque page qui est une section d'en-tête pour le menu et la méthode changePage() pour passer d'une page à l'autre dans un select.

Avec le contenu de l'attribut option qui représente la valeur à envoyer au formulaire lorsque l'option est sélectionnée. L'attribut value, indique ce qui est envoyé au serveur lorsque le formulaire est saisi.

```
<header>
  <h1>Marseille Météo</h1>
  <nav>
    <a href="home">Météo du jour</a>
    <a href="iframe">Générer l'iFrame</a>
    <a href="documentation_api">Documentation API</a>
    <a href="alentour">Météo des alentours</a>
  </nav>
  <select id="select" onchange="changePage()">
    <option class = "option" selected>Menu</option>
    <option class = "option" value="home" >Météo du jour</option>
    <option class = "option" value="iframe">Générer l'iFrame</option>
    <option class = "option" value="documentation_api">Documentation API</option>
    <option class = "option" value="alentour">Météo des alentours</option>
  </select>
</header>
```

Tout ce qu'il y a à faire, c'est de taper un bout de code :

```
@section('header')
```

Cela diffère beaucoup d'un fichier .html dont l'en-tête est représenté de cette manière : <head>

```
<meta charset="utf-8" name="viewport" content="width=device-width, initial-scale=1.0">
  <title> Documentation De L'api</title>
  <link rel="stylesheet" href="C:\laragon\www\meteomarseille\documentation_api\documentation_api.css">
</head>
<body>
  <nav role="navigation" id="topnav_responsive_menu">
<ul>
<li><a href="">Météo du jour</a></li>
<li><a href=""> Générer L'iFrame </a></li>
<li><a href="">Documentation De L'API</a></li>
</ul>
```

Le résultat obtenu dans toutes les pages où la section 'header' a été ajouté a été représenté de la manière suivante :



```
header nav {
  background-color: #59C6F2;
  color: white;
  display: flex;
  justify-content: space-evenly;
  padding: 15px;
  font-weight: bold;
  position: sticky;
  top: 0;
}

header h1 {
```

```
header a {
  font-size: 18px;
}
```

Bien entendu, le résultat ne va pas sans le code css que nous avons ajouté pour le 'nav' (menu) :

'nav' représente le menu avec les différentes options de la page.

'h1' représente le logo de notre site "Marseille Météo" en haut de chaque page.

'a' représente la taille des différents noms de pages.

Nous avons alors le conteneur de la page, une div qui ne se ferme qu'à la toute fin de la page.
C'est dans cette div que nous allons placer tout le contenu de la page.

```
<Div class="docapi">
</Div>
```

```
.docapi {
  color: #fff;
  text-align: center;
  display: flex;
  padding: 15px;
  border-radius: 10px;
  flex-direction: column;
  background-color: #59c7f2d8;
}
```

Pour le côté visuel, en ajoutant le css nous avons fait un bloc transparent pour mettre en valeur la vue de marseille mais tout en gardant la lisibilité pour le demandeur.

Le résultat du bloc est le suivant, dans cet extrait :



Je vais maintenant introduire le contenu dans le bloc api.

Tout d'abord, j'ai créé une courte phrase pour montrer le point positif principal de l'api. Avec une classe dans le p, de sorte que l'on puisse la modifier en css ou l'appeler lors d'une commande javascript à notre guise. De même pour le bouton qui redirige l'utilisateur vers l'api.

```
<div>
  <p class="title">Documentation De L'API
  <p>
  <p class="text">L'API permet d'accéder à l'ensemble des données disponibles sur la plateforme de <br>
    | manière cohérente et hiérarchisée.</p>
  <p class="text-t2">L'API permet d'accéder à l'ensemble des données<br> disponibles sur la plateforme de <br>
    | manière cohérente et hiérarchisée.</p>

  <button class="ApiReference" type="button"><a href="http://sdlapp2.dev.mars:8191/api/now">API Référence</a></button>
</div>
```

Voici le rendu final sur la page.



J'ai ensuite utilisé une petite phrase d'accroche pour énoncer les raisons d'utiliser notre api, ainsi que les différents temps que nous allions enregistrer dans une table avec comme class "my-table".

```

<h4 class="text"> Plus de 2 ans d'informations météo. </h4>
<table class="my-table">
  <thead>
    <tr>
      <th>Ensoleillé</th>
      <th>Couvert</th>
      <th>Nuageux</th>
      <th>Pluie</th>
      <th>Orage</th>
      <th>Neige</th>
      <th>Brouillard</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td></td>
      <td></td>
      <td></td>
      <td></td>
      <td></td>
      <td></td>
      <td></td>
    </tr>
  </tbody>
</table>

```

```

.my-table {
  width: 100%;
  border-collapse: collapse;
}

.my-table th,
.my-table td {
  border: 2px solid #ffff;
  padding: 15px;
  text-align: center;
}

.my-table th {
  background-color: #ffff;
  vertical-align: bottom;
  color: #59c7f2;
}

```

En css, on peut modifier le tableau avec sa classe, qui s'écrit en css avec un . puis le nom de la classe. On peut choisir sa taille, les différents niveaux de caractéristiques pour qu'il soit de la même taille. Les différents écarts de remplissage sur les quatre côtés des bords blancs. L'alignement des images, etc...

Voici le rendu final sur la page.



Ensuite la réalisation du tableau explicatif des caractéristiques de l'api avec comme nom de classe "documentation_info_api".

```


| Information | Détail                                                                                                                                                                                                   |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| coord       | <i>lon</i> : La longitude nous renseigne sur sa position est-ouest.<br><i>lat</i> : La latitude nous renseigne sur sa position nord-sud.                                                                 |
| weather     | <i>id</i> : L'identifiant unique du type de temps.<br><i>libel</i> : La description ou le libellé du type de temps.<br><i>icon</i> : L'icône associée au type de temps pour une représentation visuelle. |
| main        | <i>temp</i> : La température relevée.                                                                                                                                                                    |
| wind        | <i>speed</i> : La vitesse du vent.                                                                                                                                                                       |
| rain        | <i>precipitation_rain</i> : La quantité de pluie tombée.                                                                                                                                                 |
| snow        | <i>precipitation_snow</i> : La quantité de neige tombée.                                                                                                                                                 |
| name        | Le nom de la localisation où les relevés météorologiques ont été effectués.                                                                                                                              |
| date        | Date de la météo actuelle, ou de celle souhaitée par l'utilisateur, entrée par celui-ci dans l'URL.                                                                                                      |


```

Avec son code css pour la partie esthétique, mais aussi lorsque la souris est placée sur l'une des caractéristiques, elle change de couleur grâce à "hover": .documentation_info_api tbody tr:hover { background-color: #59a7e2;

```

.documentation_info_api {
    width: 100%;
    border-collapse: collapse;
    margin: 20px 0;
    font-size: 18px;
    text-align: left;
}

.documentation_info_api thead tr {
    background-color: #FFA000;
    color: #ffffff;
    text-align: left;
    font-weight: bold;
}

.documentation_info_api th,
.documentation_info_api td {
    padding: 12px 15px;
}

.documentation_info_api tbody tr {
    border-bottom: 1px solid #a8d3ef;
}

.documentation_info_api tbody tr td {
    border: 2px solid #FFA000;
}

.documentation_info_api tbody tr:hover {
    background-color: #59a7e2;
}

```

Voici le rendu final sur la page.

Information	Détail
coord	'lon' : La longitude nous renseigne sur sa position est-ouest. 'lat' : La latitude nous renseigne sur sa position nord-sud.
weather	'id' : L'identifiant unique du type de temps. 'libel' : La description ou le libellé du type de temps. 'icon' : L'icône associée au type de temps pour une représentation visuelle.
main	'temp' : La température relevée.
wind	'speed' : La vitesse du vent. 'deg' : L'orientation du vent en degrés.
rain	'precipitation_rain' : La quantité de pluie tombée.
snow	'precipitation_snow' : La quantité de neige tombée.
name	Le nom de la localisation où les relevés météorologiques ont été effectués.
date	Date de la météo actuelle, ou de celle souhaitée par l'utilisateur, entrée par celui-ci dans l'URL.

Et la dernière partie de ma page était la démonstration de la configuration de l'url. Dans cette partie, lorsque l'utilisateur appuie sur l'un des deux boutons, un exemple d'url de l'api apparaît pour permettre au développeur de définir ses conditions.

prenant, l'exemple que l'utilisateur appuie, sur la première option :

Récuperer les données du temps d'aujourd'hui



ce bout de code prend alors forme :

```
<div>
    <span class="aide_recherche"></span>
    <button class="btn-donate" onclick="updateText1()">Récuperer les données du temps
d'aujourd'hui
    </button>
</div>
```

Grâce à "updateText1", le script javascript suivant prend forme. Il modifiera la div avec la classe .bloc_url `<p class="bloc_url">Exemple...</p>` avec le texte que j'ai mis dans le textContent. Et pour une lisibilité absolue dans la div bloc_api, `<p class="bloc_api"></p>`

`<p class="text">Cliquez sur une des options ci-dessous pour avoir un exemple d'URL : </p>`
nous changeons le texte pour qu'il explique l'utilité de l'url.

```
<script>    function updateText1() {
```

```

document.querySelector('.bloc_url').textContent =
'http://sdlapp2.dev.\nmars:8191/api/now';
document.querySelector('.bloc_api').textContent = "→ Donne l'API de la météo
actuelle.";
</script>

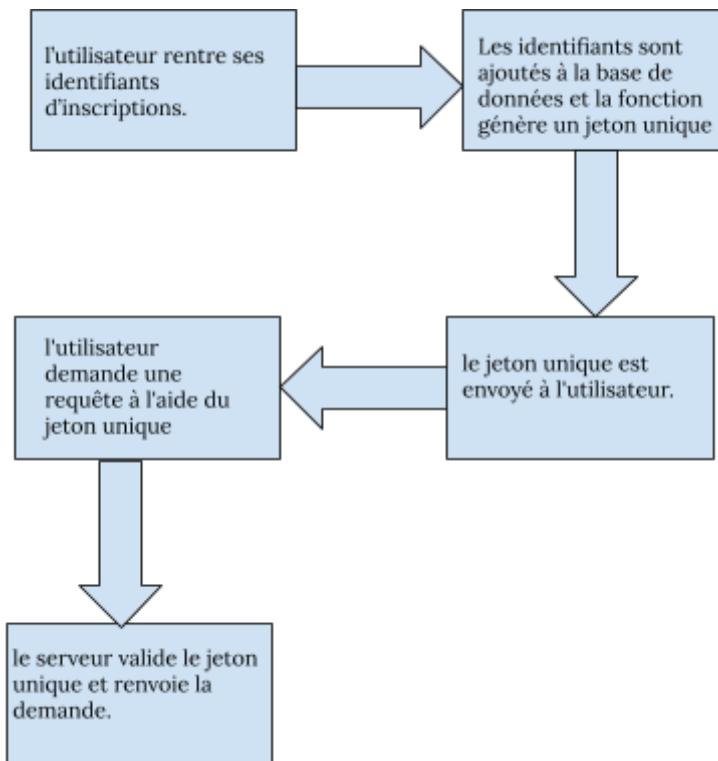
```

Voici le rendu final sur la page.



Pour la partie sécurité, j'ai pu faire un token d'authentification.

En Laravel, un token d'authentification est souvent utilisé pour gérer l'accès sécurisé aux API. Laravel utilise principalement les JSON Web Tokens (JWT) pour implémenter l'authentification par token. Voici comment cela fonctionne de manière concise :



Lorsqu'un utilisateur se connecte avec succès, Laravel génère un jeton unique et le renvoie au client. Ce token est ensuite utilisé par le client pour toutes les requêtes API suivantes en l'incluant dans les en-têtes HTTP. Le

serveur Laravel valide ce token pour chaque requête entrante pour assurer que l'utilisateur est authentifié. Les tokens peuvent expirer et nécessitent alors une réauthentification pour obtenir un nouveau token.

Pour ce faire, j'ai d'abord créé une table sur laravel :

```
public function up(): void
{
    Schema::create('personal_access_tokens', function (Blueprint $table) {
        $table->id();
        $table->morphs('tokenable');
        $table->string('name');
        $table->string('token', 64)->unique();
        $table->text('abilities')->nullable();
        $table->timestamp('last_used_at')->nullable();
        $table->timestamp('expires_at')->nullable();
        $table->timestamps();
    });
}
```

Puis j'ai utilisé la commande migrate.

```
php artisan migrate
```

La migration de base de données est le processus de transfert de données d'une base de données source vers une base de données sans cible. Tous ces éléments ont ensuite été ajoutés à la base de données.

	→	id	tokenable_type	tokenable_id	name	token	abilities	last_used_at	expires_at	created_at	updated_at	
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	App\Models\User	78 auth_token 7c9fd8a5a64db2fc982a9832dc45e50cd1c2f75ac8d239225... [""]	NULL	NULL	2024-06-14 12:05:22	2024-06-14 12:05:22		
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	App\Models\User	78 auth_token 13e8c94fd45408c4e2485e167c4d81c5475ea991d02df3771... [""]	NULL	NULL	2024-06-14 12:11:55	2024-06-14 12:11:55		
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	App\Models\User	96 auth_token 925fb264815d1a30286f29cb7fe5051bad18a1a3cde9dc5522... [""]	NULL	NULL	2024-06-14 13:19:25	2024-06-14 13:19:25		
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	App\Models\User	97 auth_token b08f7cd47eb90c1e1165ac50beee7a2992052f329200f18a46... [""]	NULL	NULL	2024-06-18 07:07:41	2024-06-18 07:07:41		
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	App\Models\User	98 auth_token 2bd4beb2123dcf88bd0a4c1daba479f08eac2584dfa1f88d... [""]	NULL	NULL	2024-06-18 07:48:52	2024-06-18 07:48:52		
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	App\Models\User	99 auth_token a763ce17ceb596fce0ec59b0e621527adbd5b4102f23603af... [""]	NULL	NULL	2024-06-18 07:51:20	2024-06-18 07:51:20		
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	App\Models\User	100 auth_token 972c96fd467795c8144e6014a07559fc53d70af9239176cb8e... [""]	NULL	NULL	2024-06-21 08:56:29	2024-06-21 08:56:29		

J'ai ensuite ajouté deux pages pour l'utilisateur, une page d'inscription et une page de connexion. C'est lorsqu'il s'inscrit, un token lui est attribué. Et elle s'ajoute à la base de données.

```
if($test_email == "yes") {
    session()->flash('app_success', 'Votre inscription a bien été enregistré, veuillez vous connecter.');

    $user = $this->create($request->all());

    auth()->login($user);
    $token = $user->createToken('auth_token')->plainTextToken;
    dd($token);

    return response()->json([
        'access_token' => $token,
        'token_type' => 'Bearer',
    ],200);
}
```

The screenshot shows a registration form titled "Inscription". It includes fields for "Nom" (Name), "Email" (with value "SaraMarie@curie.mr"), "Password", and "Confirm". At the bottom, there is a "S'enregistrer" (Register) button and a link "Vous avez déjà un compte ?" (Do you have an account?).

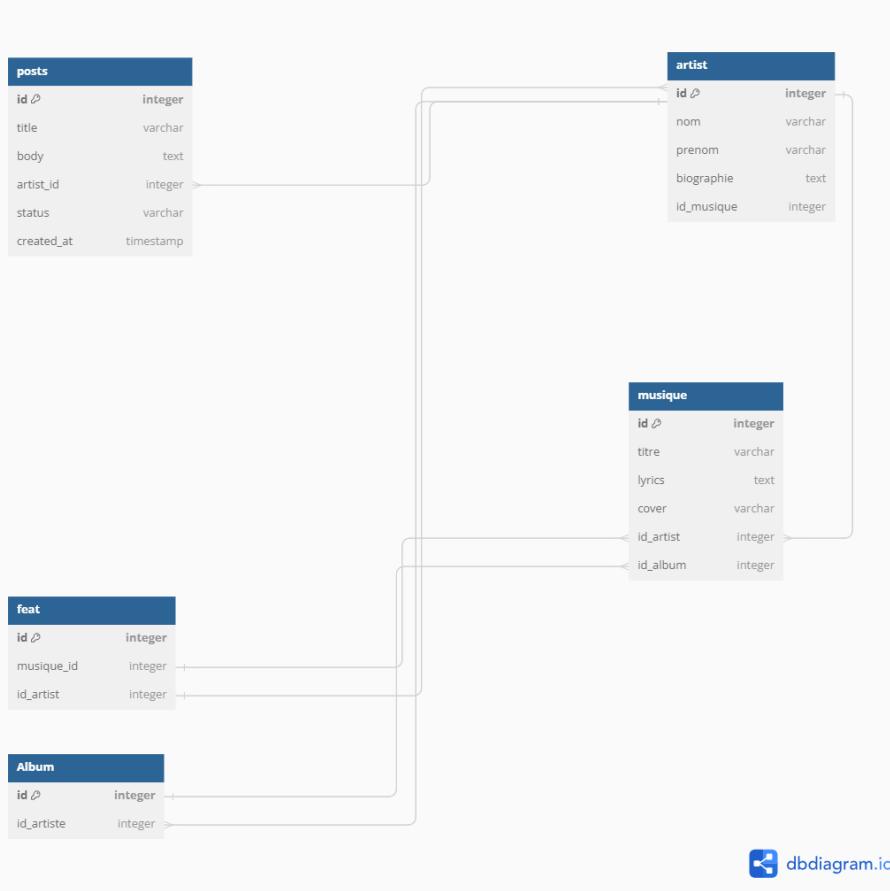
```
"8|kz0gOpVj1FbB3Mguh41YLcaz5ks0ddeSBYjSgDLXbf729727" // app\Http\Controllers\Auth\RegisterController.php:34
```

<input type="checkbox"/>	Éditer	Copier	Supprimer	8	App\Models\User	101 auth_token 9775573dad079f6e361a302ced46bd8f9aa6bd4d5876bd3398... [""]	NULL	NULL	2024-06-25 14:04:04	2024-06-25 14:04:04
--------------------------	--------	--------	-----------	---	-----------------	---	------	------	---------------------	---------------------

et lorsque l'utilisateur se déconnecte, le token est supprimé.

```
public function logout(Request $request)
{
    $request->user()->currentAccessToken()->delete();
    return response()->json(['message'=> 'Token revoked'], 200);
}
```

C'est le plus gros projet que nous ayons réalisé ensemble. Pour le premier projet, nous avons principalement appris le html, css, javascript, php, et le laravel. Mettre en place un modèle sql pour les tables dont nous avons besoin pour construire notre site.



À créer des routes php. Voici un exemple de route pour la partie Album.

```
Route::resource('Album', 'AlbumController');
```

Comprendre les deux méthodes de requêtes get et post:

```
Route::get('login', [LoginInfoController::class, 'showLoginForm'])->name('login');
Route::post('login', [LoginInfoController::class, 'login']);
```

À utiliser des fonctions dans les contrôleurs pour afficher différentes activités.

```
public function index(Index $request)
{
    $search = $request->input('q');
    if($search!="") {
        $columns = \Schema::getColumnListing('album');
        $datas = Album::where(function ($query) use ($search,$columns) {
            foreach($columns as $column) {
                $query->orWhere($column, 'LIKE', '%' . $search . '%');
            }
        })->paginate(10);
        $datas->appends(['q' => $search]);
    }
}
```

```

else{
    $datas = Album::paginate(10);
}
return view('pages.album.index')->with("records",$datas)->with("q",$search);

```

la manière de rendre nos sites Web responsives en fonction des dimensions dans inspecter.
Dans ces code, voici la différence entre le menu dans différentes dimensions :

The screenshot shows a mobile device displaying a weather website for Marseille. At the top, there's a header with "Marseille Météo" and a "Menu" button. Below the header is a large image of the Marseille skyline. On the left, there's a sidebar with "Météo de Marseille" and a forecast card for "Marseille - 2024-06-27 15:00:00" showing a temperature of 29°C and a sun icon. A dropdown menu is open, listing "Menu", "Météo du jour", "Générer l'iFrame", "Documentation API", and "Météo des alentours". To the right of the menu, there's a weather icon showing a sun and a cloud. The bottom of the screen shows a navigation bar with "HOME", "SHOP", "SCHEDULE", and "DISCOGRAPHY".

Dimensions: iPhone SE ▾ 375 × 667

```

@media screen and (max-width: 500px) {
    header nav {
        display: none;
    }

    section h2 {
        margin-right: 0px;
    }

    nav div {
        padding: 5px;
    }

    .vitesse_vent {
        display: none;
    }
}

```

Dans le premier projet que j'ai réalisé, j'ai également pu modifier le menu, mais au lieu de créer plusieurs supports, j'ai simplement spécifié que si la dimension de l'utilisateur était supérieure à une certaine taille, le menu devait être modifié :

The screenshot shows a mobile device displaying a music artist's website for "NEOLESS". The top navigation bar includes links for "HOME", "SHOP", "SCHEDULE", and "DISCOGRAPHY". Below the navigation is a large image of a person wearing a black beanie. At the bottom of the screen, there's a call-to-action button with the text "...LISTEN NOW...".

```

@media screen and (max-width: 1024px) {
    .topnav {
        background-color: #333;
        display: flex;
        align-items: center;
        width: 100%;
    }

    .topnav_link {
        color: white;
        padding: 12px;
        text-decoration: none;
    }

    .topnav_link:hover {
        color: #0078b4;
    }

    /* hide responsive menu */
    #topnav_hamburger_icon,
    #topnav_responsive_menu {
        display: none;
    }

    #topnav_responsive_menu ul {
        display: flex;
        flex-wrap: wrap;
        flex-direction: column;
        align-content: center;
        position: absolute;
    }
}

```

```

margin: 0;
right: 0;
top: 0;

min-width: 103vw;
height: 100vh;
padding: 56px 0 0;

text-align: center;
font-size: xxx-large;
background: #000000;
list-style-type: none;
-webkit-font-smoothing: antialiased;
}

#topnav_responsive_menu li {
  padding: 12px 24px;
}

#topnav_responsive_menu a {
  white-space: nowrap;
  color: #ffffff;
  text-decoration: none;
}

/* And let's slide it in from the right */
#topnav_responsive_menu.open {
  transform: none;
  position: fixed;
}

```

des modèles utilisés pour interagir avec les tables. exemple de code pour la table Album.

```

@property int $id id
@property varchar $nom nom
@property varchar $bio bio
@property int $id_artiste id artiste
@property timestamp $created_at created at
@property timestamp $updated_at updated at
@property IdArtiste $artist belongsTo
@property \Illuminate\Database\Eloquent\Collection $musique hasMany

*/
class Album extends Model
{
    /**
     * Database table name
     */
    protected $table = 'Album';

    /**
     * Mass assignable columns
     */
    protected $fillable=['id', 'id_artiste',
    'nom',
    'bio'];
}

```

établir une page de connexion avec les vérifications et l'ajouter dans une base de données.

```

public function register(Request $request)
{
$validatedData = $request->validate([
'name' => 'required|string|max:255',
'email' => 'required|string|email|max:255|unique:users',
'password' => 'required|string|min:8',
]);
$user = User::create([

```

```
'name' => $validatedData['name'],
'email' => $validatedData['email'],
'password' => Hash::make($validatedData['password']),
);
```

Et beaucoup d'autres choses que je ne peux malheureusement pas détailler.

4) Compétences professionnelles mises en œuvre

Lors de mon stage à la division développement logiciel dans le cadre de mon BTS SIO, j'ai mis en œuvre diverses compétences professionnelles essentielles. J'ai développé des applications web dynamiques en utilisant des langages comme HTML, CSS, JavaScript, et PHP, tout en intégrant des bases de données MySQL pour la gestion des données.

J'ai également travaillé avec le framework Laravel pour améliorer l'efficacité et la structure de mes projets. Mais j'ai également développé des compétences organisationnelles cruciales. J'ai appris à gérer mon temps efficacement en planifiant et en priorisant les tâches grâce à des outils comme TortoiseSVN. J'ai participé à des réunions d'équipe régulières pour assurer une communication fluide et une coordination optimale des projets. J'ai également appliqué des méthodes qui m'ont permis de travailler en sprints, d'adapter rapidement les projets aux changements. Et organiser des réunions avec le demandeur pour présenter le projet oralement.

Enfin, j'ai appliqué les principes de sécurité web pour protéger les requêtes contre les vulnérabilités courantes.

1) Bilan sur les projet

Projet 1 : Site de Musique

Pour ce projet, j'ai développé un site de musique pour mon frère en utilisant le framework Laravel. Ce projet m'a permis de maîtriser l'utilisation de Laravel pour la gestion des bases de données, la création de routes et de contrôleurs, mais aussi d'approfondir les langages de programmation html, css, javascript, et php.

Projet 2 : Site de Météo Marseille

Le deuxième projet consistait à créer un site de météo pour la ville de Marseille, réalisé en collaboration avec deux autres stagiaires et avec l'aide de mes tuteurs de stage. Ce site utilise un iframe et des APIs externes pour obtenir des données météorologiques en temps réel et les afficher de manière intuitive. Nous avons également implémenté une fonctionnalité d'authentification par token pour sécuriser l'accès aux informations. Ce projet m'a permis de travailler en équipe en utilisant les méthodologies Agile et d'approfondir mes compétences en gestion des tokens d'authentification et en intégration d'APIs.

2) Bilan personnel

Ces projets m'ont permis de développer significativement mes compétences techniques en développement web, en particulier avec Laravel. J'ai acquis une compréhension approfondie des frameworks PHP, de l'intégration d'APIs et de la sécurité des applications web. Au-delà des aspects techniques, j'ai également amélioré mes compétences organisationnelles et collaboratives. Travailler en équipe sur le projet de météo m'a appris à communiquer efficacement, à partager les responsabilités et à respecter les délais. Les conseils de mes tuteurs de stage ont été précieux pour affiner mes compétences et améliorer la qualité de mon code.

En somme, ces expériences m'ont préparé à affronter des défis professionnels plus complexes et m'ont donné une base solide pour ma future carrière en développement web.

3) Références :

framework : En programmation informatique, le terme « framework » se traduit par « structure logicielle ». C'est est un ensemble cohérent de composants logiciels structurels qui sert à créer les fondations ainsi que les grandes lignes de tout ou partie d'un logiciel, c'est-à-dire une architecture.

Injection de dépendances : C'est une technique de programmation où un objet (comme une classe) reçoit ses dépendances (autres objets dont il a besoin pour fonctionner) de l'extérieur plutôt que de les créer lui-même. Cela rend le code plus flexible et plus facile à tester.

Couche d'abstraction de base de données expressive : C'est une couche de code entre l'application et la base de données qui simplifie les interactions avec la base de données. Elle permet aux développeurs d'écrire des requêtes de manière plus claire et plus compréhensible, sans se soucier des détails techniques de la base de données.

Tests unitaires : Ce sont des tests automatisés qui vérifient que chaque petite partie de votre code (comme une fonction ou une méthode) fonctionne correctement individuellement.

Tests d'intégration: Ce sont des tests automatisés qui vérifient que différentes parties de votre code fonctionnent bien ensemble.

Conception graphique vectorielle : C'est la création d'images en utilisant des formes géométriques comme des points, des lignes et des courbes. Ces images peuvent être agrandies ou réduites sans perdre en qualité.

Prestataires : Ce sont des entreprises ou des individus qui fournissent des services ou des produits à d'autres entreprises ou individus. Par exemple, une entreprise de nettoyage est un prestataire de services de nettoyage.

Requête API : C'est une demande envoyée à une Application Programming Interface (API), qui est une sorte de messager permettant à différents logiciels de se parler. Par exemple, quand une application météo demande les dernières informations météorologiques à un serveur, elle envoie une requête API.

Iframe : C'est une balise HTML qui permet d'intégrer une autre page web à l'intérieur d'une page web. Par exemple, vous pouvez avoir une vidéo YouTube intégrée dans une page web grâce à un iframe.

JSON Web Tokens (JWT) : Ce sont des jetons de sécurité utilisés pour échanger des informations de manière sécurisée entre deux parties. Ils contiennent des données qui peuvent être vérifiées et sont souvent utilisées pour gérer les sessions utilisateur sur le web.

HTTP (Hypertext Transfer Protocol) : C'est le protocole utilisé pour transférer des pages web sur Internet. Quand vous visitez un site web, votre navigateur utilise HTTP pour demander la page web au serveur et pour recevoir le contenu en retour.