

# Rapport de Stage

Entreprise : SOCOREL - REDON



Nos expertises ▼

Nos Produits ▼

Nos références

✉ Contact



## Conception

Socorel est spécialisé dans la **recherche et développement** de produits innovants pour la **gestion de l'énergie et des fluides**.



## Réparation

**25 ans d'expérience dans la réparation professionnelle** nous permet de répondre à vos besoins d'assistance et maintenance.



## Nos produits

Retrouvez notre **gamme de produits GTB** (Gestion Technique des Bâtiments) conçus pour **compléter et ouvrir les capacités** des automates standards modbus.

# SOMMAIRE

## **Sommaire**

I. PRÉSENTATION DE L'ENTREPRISE.....	3
Introduction.....	3
Historique et Contexte.....	3
Activités et Savoir-Faire.....	3
Compétences et Expertise.....	3
Conclusion.....	3
II. REMERCIEMENT.....	4
III. OBJECTIF DE LA MISSION :.....	4
IV. Présentation et déroulé de ma mission.....	5
V. Avancement et progression de la mission.....	6
1. Semaine 1 : du 06/01/2025 au 10/01/2025.....	6
2. Semaine 2 : du 13/01/2025 au 17/01/2025.....	7
3. Semaine 3 : du 20/01/2025 au 24/01/2025.....	10
4. Semaine 4 : du 27/01/2025 au 31/01/2025.....	12
5. Semaine 5 : du 03/02/2025 au 07/02/2025.....	13
6. Semaine 6 : du 10/02/2025 au 14/02/2025.....	15
VI. Ce que je retiens de mon stage :.....	16
VII. Veilles technologiques.....	17
VIII. ANNEXES.....	19

# I. PRÉSENTATION DE L'ENTREPRISE

## Introduction

SOCOREL est une entreprise française située à Redon, en Bretagne, spécialisée dans la conception, le développement et la réparation d'équipements électroniques. Elle propose des solutions innovantes pour la gestion de l'énergie et des fluides dans les bâtiments, visant à optimiser leur efficacité et réduire leur consommation énergétique.

## Historique et Contexte

Fondée en 2013, SOCOREL s'est imposée comme un acteur clé de l'électronique industrielle. Son objectif est d'apporter des solutions performantes et durables aux entreprises et collectivités, en mettant l'accent sur l'innovation et la qualité des produits fabriqués en France.

## Activités et Savoir-Faire

L'entreprise intervient principalement dans :

- **Conception et fabrication** : Développement de systèmes électroniques pour la Gestion Technique des Bâtiments (GTB).
- **Maintenance et réparation** : Diagnostic et remise en état des équipements électroniques pour prolonger leur durée de vie.
- **Optimisation énergétique** : Solutions permettant une gestion efficace des infrastructures et une réduction des coûts énergétiques.

## Compétences et Expertise

SOCOREL s'appuie sur un large éventail de compétences :

- **Électronique industrielle** : Conception et fabrication de circuits électroniques.
- **Programmation embarquée** : Développement de logiciels intégrés pour l'automatisation des équipements.
- **Contrôle qualité et innovation** : Tests rigoureux et adaptation continue aux évolutions technologiques.

## Conclusion

Grâce à son expertise et son engagement en faveur de solutions durables, SOCOREL joue un rôle clé dans l'optimisation énergétique et la maintenance des équipements électroniques. Son savoir-faire permet d'accompagner les entreprises et collectivités dans l'amélioration de leurs infrastructures, contribuant ainsi à une gestion plus efficace et responsable de l'énergie.

## II. REMERCIEMENT

Je tiens à exprimer mes sincères remerciements à Monsieur Hemery Vincent, dirigeant de SOCOREL, pour m'avoir offert l'opportunité de réaliser mon stage au sein de l'entreprise. Cette expérience m'a permis de renforcer et d'enrichir mes compétences en développement web dans un environnement professionnel.

Je souhaite également remercier chaleureusement Monsieur Lanuel Sullivan, collaborateur chez SOCOREL, avec qui j'ai eu le plaisir de présenter l'avancée de ma mission, de discuter des possibilités d'ajouts ou d'améliorations des fonctionnalités, et d'échanger sur les différents enjeux du projet.

Enfin, je tiens à adresser mes remerciements à Monsieur Dederck Laurent, soudeur chez SOCOREL, qui, face à une certaine difficulté technique rencontrée, m'a proposé une idée précieuse, me permettant de surmonter cette obstacles et de trouver une solution efficace.

## III. OBJECTIF DE LA MISSION :

Lors de mon stage, j'ai travaillé sur le développement d'un outil de gestion des entrées de stock et de suivi des produits. Cet outil, conçu comme une solution web, permet de créer des numéros de série pour chaque produit et de les suivre tout au long de leur cycle de vie, depuis leur enregistrement dans la base de données jusqu'à leur livraison à la logistique du distributeur. L'objectif principal était d'améliorer la traçabilité des produits, d'optimiser la gestion des stocks et de faciliter le processus logistique. Ce projet a impliqué la conception de l'interface utilisateur, l'intégration de bases de données, ainsi que la mise en place de fonctionnalités pour la gestion et la recherche des produits par numéro de série.

## IV. Présentation et déroulé de ma mission

Date de réalisation de la mission : Du 06/01/2024 au 14/01/2024

### Outil et langage utilisé pour réaliser l'outil WEB :

- **WAMP** : Un environnement de développement local qui combine Windows, Apache, MySQL, et PHP. WAMP permet d'héberger un serveur web localement sur une machine Windows, facilitant le test et le développement d'applications web sans avoir besoin d'un serveur distant. Apache gère les requêtes HTTP, MySQL s'occupe de la gestion des bases de données, et PHP génère des pages dynamiques côté serveur.
- **Visual Studio Code (VS Code)** : Un éditeur de code léger mais puissant, utilisé pour écrire du code dans divers langages comme PHP, HTML, CSS et JavaScript. VS Code offre des fonctionnalités avancées telles que l'autocomplétion, le débogage intégré, et des extensions pour améliorer l'efficacité du développement, tout en permettant une gestion facile des projets et des fichiers.
- **MySQL** : Un système de gestion de base de données relationnelle utilisé pour stocker, organiser et manipuler les données de l'application. MySQL est particulièrement adapté aux applications web dynamiques, permettant de gérer de grandes quantités de données et d'assurer des connexions rapides et fiables avec le backend via PHP.
- **PHP** : Un langage de programmation côté serveur qui génère des pages web dynamiques en fonction des actions de l'utilisateur, comme la soumission de formulaires. PHP permet de gérer la logique métier, de traiter les données envoyées par l'utilisateur, et de se connecter à la base de données MySQL pour récupérer ou insérer des données.
- **JavaScript** : Un langage de programmation côté client utilisé pour rendre l'interface utilisateur interactive. Il permet de manipuler le DOM (Document Object Model) pour changer dynamiquement le contenu de la page sans avoir à la recharger. JavaScript est également utilisé pour implémenter des fonctionnalités comme la gestion des événements utilisateurs, la validation de formulaires, et l'appel de scripts côté serveur (AJAX).
- **HTML et CSS** : HTML (HyperText Markup Language) est utilisé pour définir la structure des pages web, tandis que CSS (Cascading Style Sheets) est utilisé pour styliser ces pages (mise en page, couleurs, polices, etc.). HTML est la base de la structure du contenu, tandis que CSS définit l'apparence visuelle et l'agencement des éléments.

## V. Avancement et progression de la mission

### 1. Semaine 1 : du 06/01/2025 au 10/01/2025

Au cours de la première semaine de mon stage, j'ai d'abord réfléchi aux fonctionnalités principales à implémenter dans l'outil de gestion. En l'absence d'un cahier des charges précis, j'ai pris l'initiative de définir un cahier des charges de base pour guider l'avancement de mon projet.

J'ai commencé par concevoir les différentes bases de données nécessaires, notamment la table "products", qui permettra d'enregistrer les produits avec les informations suivantes :

id	barcode	nom_produit	code_type	created_at	famille	commercialise
26	1	FP8-RS/24V	1015	2025-02-17 09:50:19	FP8	1

- **barcode** : numéro de série unique pour chaque produit, permettant de l'identifier.
- **nom\_produit** : désignation du nom du produit.
- **code\_type** : identifie un type spécifique de produit ; par exemple, les produits FP8-RS/24V ont le code produit 1015.
- **created\_at** : date d'enregistrement du produit dans la base de données.
- **famille** : indique la famille à laquelle appartient le produit, permettant de regrouper des sous-types sous une même catégorie (ex : la famille FP8 regroupe plusieurs déclinaisons de produits).
- **commercialise** : précise si une gamme de produits, définie par le code\_type, est encore commercialisée ou non.

Une fois la table "products" créée, j'ai pu développer ma première page : "produit.php" (voir annexe 1). Cette page me permet d'entrer le nom et la famille d'un produit. Lors de la soumission du formulaire, elle fait appel au fichier "process.php" pour enregistrer le nouveau produit dans la base de données.

Dans un second temps, j'ai créé la page d'affichage des différents produits enregistré dans la base de données : "vue.php" (voir annexe 2), qui générerait de façon dynamique un tableau listant l'ensemble des produits avec les caractéristiques des produits comme défini dans la base de donnée, tout en affichant un système de filtre afin de sélectionner et d'afficher une famille bien précise ou juste afficher un seul type de produit en fonction de son nom.

Une fonction Gérer la commercialisation était définie pour modifier le statut de la donnée commercialise (représenté dans la colonne commercialisation de l'annexe 2) et qui permettait à l'utilisateur de changer le statut de commercialisation d'une gamme de produit (défini par la donnée code\_type) qui une fois défini sur 0, empêchait l'enregistrement d'un nouveau produit de cette gamme dans la base de données.

## 2. Semaine 2 : du 13/01/2025 au 17/01/2025

Pour démarrer la deuxième semaine de développement, j'ai apporté une modification importante à ma page «produit.php», en améliorant la gestion de l'enregistrement des produits. J'ai mis en place un nouveau système qui associe chaque produit à un code de type spécifique. Cette association est effectuée grâce à un dictionnaire (\$productMapping) qui fait correspondre les noms des produits à leurs codes de type respectifs, afin de résoudre les problèmes de correspondance rencontrés dans la première version du code.

```
// Correspondance entre nom_produit et code_type
$productMapping = [
    "FP8-RS/24V" => 1018,
    "FP8-RS/230V" => 1019,
    "FP8-IP/24V" => 1020,
    "FP8-IP/230V" => 1021,
    "Teleinfo-RS/24V" => 1022,
    "Teleinfo-RS/230V" => 1023,
    "Teleinfo-IP/24V" => 1024,
    "Teleinfo-IP/230V" => 1025,
    "Sonde T+H" => 1026,
    "Sonde T,H,CO2" => 1027,
    "TIC Light-230V" => 1017,
    "R TIC" => 1028
];

// Vérifier si le produit est bien dans le mapping
if (!isset($productMapping[$nom_produit])) {
    echo json_encode(value: ["error" => "Produit inconnu. Assurez-vous qu'il est défini dans la correspondance."]);
    exit;
}

$code_type = $productMapping[$nom_produit];
```

### Explication :

1. Le dictionnaire **\$productMapping** : est un tableau associatif (ou "map") qui lie chaque nom de produit (`nom_produit`) à un identifiant unique de type (`code_type`). Par exemple, le produit "FP8-RS/24V" est associé au code 1018.

2. **La vérification** : Avant d'exécuter l'enregistrement du produit, le code vérifie si le nom du produit fourni par l'utilisateur existe dans le dictionnaire \$productMapping. Cela permet d'éviter toute erreur due à un produit inconnu. Si le produit n'est pas trouvé, un message d'erreur est renvoyé en JSON et le script est arrêté avec `exit`.
3. **Récupération du code\_type** : Si le produit est bien trouvé dans le dictionnaire, le code de type correspondant est extrait et stocké dans la variable \$code\_type. Ce code peut ensuite être utilisé pour l'enregistrement du produit dans la base de données ou toute autre opération nécessitant ce lien entre le produit et son type.

Pour finir la modification de ma page produit.php, j'ai ajouté du style, des photos des produits, et rajouter un champ de saisie afin de pouvoir enregistrer plusieurs fois le même produit en une seule fois (voir annexe 3).

Pour la fin de la semaine, j'ai créé une nouvelle table de données : "bon\_de\_commande" avec les informations suivantes :

commande_id	nom_commande	nom_produit	famille	code_type	barcode	date_creation	date_complétude	destinataire	etat_commande
235	F-PO-2501-2075	Tie Light-230V	Tie Light	1017	17-00001000	2025-01-24 10:32:32	NULL	BTIB	totale

**-commande\_id** : Correspond aux numéro de commande.

**-nom\_commande** : Correspond aux noms de la commande.

**-date\_creation** : correspond à la date d'enregistrement du bon de commande.

**-date\_complétude** : correspond à la date ou la commande a été initialisé si jamais elle n'était pas complète.

**-Destinataire** : Correspond aux destinataires de la commande (repris dans une 3ème base de données : "destinataire", qui contient les valeurs : id et nom\_destinataire).

**-etat\_commande** : données qui défini si une commande est totale ou partielle dans le cas ou il manque des produits.



J'ai ainsi fini la semaine en créant la page "bon\_commande.php", qui permet de créer une commande et l'enregistrer dans la base de données.

Enregistrer un Bon de Commande

Nom du Bon de Commande :

Destinataire :

Sélectionner un destinataire

Liste des Produits

Sélectionner	Nom du Produit	Code Type	Stock Restant	Quantité Souhaitée
<input type="checkbox"/>	R TIC	1028	4	1

Récapitulatif de la Commande

État de la Commande

☐ Totale ☐ Partielle

Confirmer la Commande

Cette page permet donc de créer et d'enregistrer un bon de commande dans la base de données, pour cela l'utilisateur doit remplir plusieurs champs dont le nom de la commande, sélectionner le destinataire dans une liste, liste qui reprend les destinataires enregistrée dans la table "destinataire" ou qui permet d'enregistrer un nouveau destinataire au travers de la fonction

`updateDestinataireHidden()`

qui va appeler un script ouvrant une pop-up pour rentrer un nouveau destinataire tout en vérifiant qu'il n'existe pas déjà ce qui engendre une erreur.

On a ensuite, la possibilité de Sélectionner des produits à inclure dans le bon de commande, avec affichage de la quantité restante en stock pour chaque produit, et contrôle de la disponibilité des produits pour éviter des dépassements de stock.

Enfin, on a la possibilité de choisir si l'état de la commande doit être totale ou partielle puis on enregistre la commande dans la base de données au travers du fichier "process\_commande.php". (annexe 4).

### 3. Semaine 3 : du 20/01/2025 au 24/01/2025

Cette semaine, j'ai commencé à travailler sur le système de génération d'étiquettes pour l'identification et le suivi des produits. L'étiquette devait inclure plusieurs éléments essentiels : le logo et l'adresse de l'entreprise, trois logos spécifiques (le marquage CE, le logo RoHS et le pictogramme de la poubelle barrée pour la collecte séparée des déchets électroniques), ainsi que le nom du produit. Elle devait également comporter deux code-barres : l'un pour le **code\_type**, qui représente la gamme du produit (son identifiant unique), et l'autre pour le numéro de série du produit.

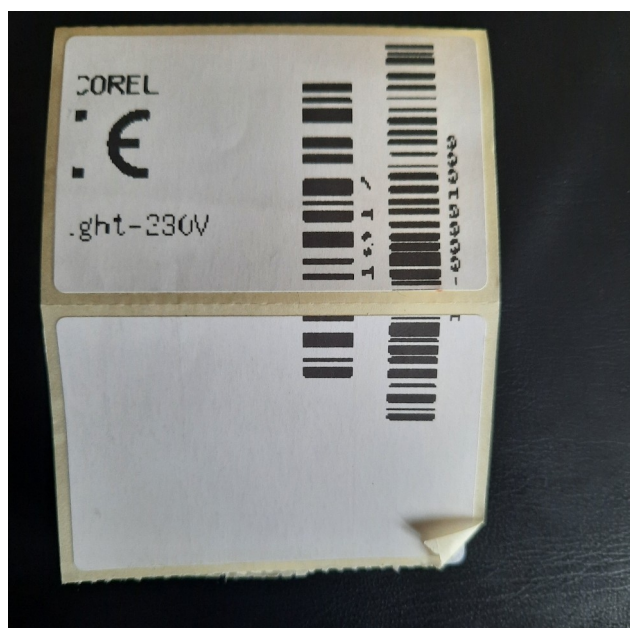
Le marquage **CE** atteste que le produit respecte les normes européennes en matière de sécurité, de santé et de protection de l'environnement. Le logo **RoHS** (Restriction of Hazardous Substances) indique que le produit est conforme à la directive européenne limitant l'utilisation de substances dangereuses, comme le plomb et le mercure, dans les équipements électriques et électroniques.

Pour cela, j'ai créé une nouvelle page de gestion des codes barres, (voir annexe 5) résumant par jour les produits qui ont été enregistré dans la base de données avec un filtre en fonction de la date du jour. La page permet ainsi d'afficher en fonction de la date sélectionné, les produits qui ont été enregistré dans la base de données avec :

- Le nom du produit
- Le code produit spécifique à un produits (donnée code\_type)
- Son numéro code barre (aussi appelé numéro de série)
- une image fusion des code-barre de code\_type et barcode (voir annexe 6 pour expliquer le processus)
- un appel de génération à la page "generate\_label.php" qui doit générer l'étiquette

J'ai ensuite commencé à travailler sur la génération de l'étiquette qui devait être imprimé sur un format 34 par 54mm mais j'ai rencontré un problème de format ou l'étiquette généré n'était pas totalement imprimé ou les différents code-barre n'étaient pas de bonne qualité pour pouvoir être scannés.

J'ai choisis de mettre cette partie de coté et la reprendre plus tard durant le stage.



Exemple d'étiquette imprimé au mauvais format, et les code-barres sont illisibles.

Enfin, pour finir la semaine, j'ai créer une nouvelle table de données, la table : "utilisateur" , qui va en fonction du groupe auquel appartient l'utilisateur, lui reteindre ou lui permettre de se rendre sur certaines pages de l'application

id	nom	email	mot_de_passe	role	groupe
1	admin	admin@gmail.com	\$2y\$10\$gxkpi65KmETGyvqzFE.Bw..u6DU2OcVc.AVDd2LUzLm...	admin	NULL

## Explication des données de la table utilisateurs

La table **utilisateurs** contient les informations suivantes :

- **id** : Identifiant unique pour chaque utilisateur, généré automatiquement.
- **nom** : Nom de l'utilisateur (obligatoire).
- **email** : Adresse email unique de l'utilisateur.
- **mot\_de\_passe** : Mot de passe de l'utilisateur, stocké de manière sécurisée (haché).
- **role** : Rôle de l'utilisateur (admin ou utilisateur).
- **groupe** : Groupe auquel l'utilisateur appartient, peut être NULL.

## Sécurisation du mot de passe

Les mots de passe sont **hachés** avec l'algorithme **bcrypt** avant d'être stockés dans la base de données. Cela les protège en cas de fuite de données, car le mot de passe ne peut pas être récupéré directement. Lors de la connexion, **password\_verify()** permet de vérifier que le mot de passe saisi correspond au haché stocké. Cette méthode sécurise efficacement les informations sensibles des utilisateurs.

#### 4. Semaine 4 : du 27/01/2025 au 31/01/2025

J'ai commencé cette nouvelle semaine, en créant une nouvelle page «historique\_commande.php», qui va récupérer toutes les commandes passées et qui va les afficher dans un tableau en affichant le nom de la commande, le destinataire, la date de création de la commande (ou de complétude si la commande n'était pas totale et que des produits ont été rajoutés), la quantité de produit dans la commande, l'état de la commande (voir annexe 7)

Après avoir créé la page historique des commandes, j'ai entrepris la mise en place de la fonctionnalité permettant d'afficher les détails de chaque commande spécifique. Pour ce faire, j'ai utilisé un paramètre de requête dans l'URL, intitulé `commande_id`, qui sert à identifier de manière unique chaque commande à afficher.

Ce paramètre *commande\_id* contient une valeur spécifique (comme par exemple «F-PO-2501-2075») qui correspond à un identifiant unique d'une commande dans la base de données. Grâce à cette valeur, la page peut récupérer les informations détaillées liées à cette commande précise et les afficher dynamiquement.

Exemple d'URL : «[http://localhost/dev\\_socorel/page/details\\_commande.php?commande\\_id=F-PO-2501-2075](http://localhost/dev_socorel/page/details_commande.php?commande_id=F-PO-2501-2075)» (voir annexe 8 pour l'exemple)

En deuxième partie de semaine, j'ai créé la page d'accueil qui permet d'accéder à la page de connexion, permettant à l'utilisateur de se connecter en entrant son nom d'utilisateur (ou adresse mail) et son mot de passe (annexe 9), une fois l'utilisateur connecté, une nouvelle zone est affichée sur l'écran résumant les pages auxquelles il a accès en fonction de son rôle (voir annexe 10)

Pour finir la semaine, j'ai modifié la page pour gérer l'état de commercialisation d'une gamme représentée par la donnée `code_type` (annexe 11) et j'ai modifié la page «détail\_commande.php» afin de pouvoir modifier le destinataire et le nom de la commande (voir annexe 12)

=> Manquement à implémenter : Modification de la quantité des produits avec possibilité de retirer une certaine quantité de produit en fonction de leur gamme

## 5. Semaine 5 : du 03/02/2025 au 07/02/2025

Cette semaine, j'ai effectué la modification la plus importante de l'ensemble du projet. En effet, j'ai dû ajouter de nouveaux éléments aux différents produits, à savoir :

- Une nouvelle version
- Un logiciel (software)
- Un matériel (hardware)

De plus, j'ai intégré la possibilité de modifier ces valeurs dans le cas où de nouvelles versions seraient créées ou mises en production.

Pour répondre à cette problématique, j'ai créé une nouvelle table de données : la table *soft-hard*.

code_type	nom_produit	hardware	software	date_mise_à_jour	version	famille
1017	Tic Light-230V	1.4	2.1	NULL	1	TIC Light
1017	Tic Light-230V	8.6	7.6	2025-02-18 11:10:00	2	TIC Light

- **code\_type** : Identifiant unique du produit, représentant sa gamme.
- **nom\_produit** : Nom attribué au produit.
- **hardware** : Version actuelle du matériel.
- **software** : Version actuelle du logiciel.
- **date\_mise\_à\_jour** : Date à laquelle le produit a été mis à jour, ou *null* pour la première version.
- **version** : Numéro de version du produit.
- **famille** : Catégorie ou groupe auquel le produit appartient.

Exemple : dans notre cas, pour le produit Tic Light-230V, on a deux version d'un même produit mais avec des caractéristiques software et hardware différentes.

Une fois la table créer, j'ai du créer une nouvelle page : «Software-hardware.php» qui reprend toutes les versions de chaque produit et les résumes dans un tableau trier par gamme de produit (voir annexe 13)

Enfin, la page «Software-hardware.php» permet aussi de créer des nouveaux produits ou aussi la possibilité de créer une nouvelle famille de produit

### Créer un nouveau produit

Code Type :	<input type="text" value="1032"/>	Nom Produit :	<input type="text" value="test"/>	Famille du Produit :	<input type="text" value="FAMILLE (Temporaire)"/>	▼
Hardware :	<input type="text" value="3.4"/>	Software :	<input type="text" value="4.5"/>	Version :	<input type="text" value="1"/>	Date de mise à jour :

18/02/2025 11:45 📅

Créer

## Ajouter une nouvelle famille

Nom de la nouvelle famille :

FAMILLE

Créer la famille

Lorsqu'une nouvelle famille est créée et qu'aucun produit ne lui est encore attribué, la valeur de la variable *famille* est enregistrée temporairement dans l'état de session. Dès qu'un produit est enregistré dans la base de données, cette variable temporaire devient une variable permanente, remplaçant ainsi sa version dans l'état de session pour devenir une véritable famille.

Enfin pour finir la semaine, j'ai modifié la page «produit.php», qui servait à enregistrer des produits dans la base de données.

Pour cette modification, j'ai retiré le dictionnaire associatifs des produits comme dans la version 2 et à la place, j'ai modifié la table «products», pour qu'elle récupère les données version, hardware, software de la table «soft\_hard» pour que lors de l'enregistrement des produits ont puisse choisir la version du produit à enregistrer. (voir annexe 14)

## 6. Semaine 6 : du 10/02/2025 au 14/02/2025

Durant la dernière semaine du stage, j'ai apporté une modification à la page «vue.php» qui servait à afficher les produits enregistrés dans la base de données afin d'y faire apparaître les données version, software et hardware d'un produit. (annexe 15)

Ensuite, je suis revenu, sur mon problème d'étiquette rencontré lors de la semaine 3.

*Rappel du problème rencontré* : Lors de l'impression des étiquettes, une partie de l'étiquette n'est pas imprimée, ou le code-barre n'est pas de bonne qualité pour pouvoir être scanné.

Après de nouveaux tests, et une frustration rencontrée, j'ai eu l'appui et la suggestion de Dederck Laurent, soudeur chez SOCOREL, qui a émis l'idée de travailler sur une étiquette avec de plus petite dimension en passant d'un format 34 par 54 mm à un format 24 par 54 mm.

Suite à cette idée, j'ai recalculé les nouvelles dimensions en pixel et modifié la position des différents éléments de l'étiquette. En modifiant les dimensions de l'image de base, j'ai fini par comprendre ce qui posait problème lors de l'impression de l'étiquette par l'imprimante : L'imprimante faisait lors de l'impression un recalcule des dimensions par rapport à ces paramètres d'où le mauvais rapport d'impression.

Une fois les dimensions de l'étiquette modifiées, j'ai effectué un test d'impression et l'imprimante à recalculer les dimensions pour s'adapter aux dimensions de l'étiquette, ce qui m'a permis d'obtenir une étiquette au bon format avec l'ensemble des éléments important à intégrer sur l'étiquette et des code-barre de bonne qualité pouvant être scanné.



Ensuite, pour continuer la semaine, j'ai créé la page de gestion des utilisateurs pour définir, modifier, créer ou supprimer des utilisateurs. Pour cela la page va récupérer les utilisateurs présents dans la base de données via la table «utilisateur» et si l'utilisateur connecté est un admin, il peut accéder à cette page et ainsi gérer les différents utilisateurs.(annexe 16)

Enfin, pour finir la semaine et ma mission, j'ai mis en place une dernière page «produit\_info\_scanner» qui permet de scanner les deux codes-barres d'un produit et d'y retourner différentes informations en fonction de ce qui est scanné : le numéro de gamme (code\_type) ou le numéro de série d'un produit (barcode) – (voir annexe 17 et 18 pour les exemples).

## VI. Ce que je retiens de mon stage :

Ce stage m'a apporté de nombreuses connaissances, notamment en matière de compétences techniques liées au développement web. J'ai particulièrement approfondi mes compétences en matière de sécurisation des données sensibles, comme les mots de passe, en mettant en place des techniques de hachage adaptées. J'ai également travaillé sur la gestion des accès aux différentes pages et fonctionnalités de l'outil, en mettant en place un système de permissions basé sur les rôles et les groupes des utilisateurs, garantissant ainsi une meilleure protection des données et une limitation des accès selon les autorisations définies.

Par ailleurs, ce stage m'a permis de renforcer mes compétences en gestion des bases de données, notamment en optimisant leur intégration et leur interaction avec les différentes pages de l'outil. J'ai appris à structurer et exploiter efficacement les données stockées afin d'améliorer la fluidité et la cohérence de l'application.



## VII. Veilles technologiques.

Pour ma veille technologiques, je me suis orienté vers deux thèmes important en ce moment car ils sont de plus en plus fréquent :

- Les fuites de données en entreprise
- Les attaques CYBER

je vais maintenant résumer quelques informations retenues lors de ma veille technologiques :

### **Fuite de données chez KIABI – Janvier 2025 :**

En janvier 2025, Kiabi a subi une fuite de données touchant près de 20 000 clients de sa plateforme "**Seconde Main by Kiabi**". Les informations compromises incluait noms, prénoms, dates de naissance, adresses postales et IBAN.

L'attaque, survenue le 7 janvier, a été réalisée via du **credential stuffing**, une technique où des hackers utilisent des identifiants volés lors d'autres fuites pour accéder aux comptes. L'attaque a été facilitée par la réutilisation des mots de passe par les utilisateurs.

En réponse, Kiabi a renforcé sa sécurité en masquant les IBAN et en imposant une réinitialisation des mots de passe. Pour prévenir ce type d'attaque, il est recommandé aux utilisateurs d'adopter des mots de passe uniques et l'authentification à deux facteurs (2FA), tandis que les entreprises doivent mettre en place des systèmes anti-bot et de détection d'anomalies.

## **Téléfonica - géant espagnol de la télécommunication – 9 janvier 2025 :**

Le **9 janvier 2025**, Telefónica, géant espagnol des télécommunications, a subi une fuite de **2,3 GB de données**, compromettant **236 493 entrées clients**, **469 724 tickets internes**, et plus de **5 000 fichiers internes** (CSV, PPTX, XLSX, DOCX, PDF, MSG).

### **Déroulement de l'attaque :**

Les hackers ont utilisé des **identifiants d'employés compromis** pour infiltrer les systèmes internes. Ils ont extrait des données de **Jira**, un outil de gestion de projets, puis les ont publiées sur un forum de hackers. Le système piraté était principalement dédié à la gestion des problèmes internes du personnel.

### **Origine des attaquants :**

Les cybercriminels, opérant sous les pseudonymes **DNA, Grep, Pryx et Rey**, ont revendiqué l'attaque. Pryx a confirmé que la cible était un système interne lié à la gestion des employés de Telefónica.

## **Attaque de PowerSchool, entreprise spécialisée dans les logiciel éducatif américains :**

En **décembre 2024**, **PowerSchool**, entreprise spécialisée dans les logiciels éducatifs, a subi une cyberattaque majeure exposant les données personnelles de **millions d'étudiants et d'éducateurs**.

### **Chronologie de l'attaque :**

- **19-23 décembre** : Début de l'accès non autorisé.
- **28 décembre** : PowerSchool découvre l'incident via son portail de support **PowerSource**.
- **7-8 janvier 2025** : Les districts scolaires et les parents sont informés.
- **13 janvier** : Communication officielle de PowerSchool.

### **Détails de la fuite :**

Les hackers ont exploité des **identifiants compromis** pour accéder à **PowerSource**, exposant des **noms, adresses, dates de naissance, numéros de sécurité sociale, informations médicales et dossiers académiques**. PowerSchool affirme qu'aucune donnée bancaire n'a été touchée.

### **Réaction de PowerSchool :**

L'entreprise a **déployé une équipe de cybersécurité**, informé les autorités et renforcé la protection des données. Elle a aussi proposé des **services de surveillance du crédit et de protection contre le vol d'identité** aux victimes.

## VIII. ANNEXES

### ANNEXE 1 : produit.php version 1 et explication de process.php:



Ajouter un Produit

Nom du Produit :

Nom du produit

Famille du Produit :

Famille du produit

Ajouter le Produit

Le script process.php, gère l'ajout d'un nouveau produit dans la base de données via une requête POST. Voici les étapes principales :

#### 1. Initialisation et validation :

- Le script inclut le fichier database.php pour se connecter à la base de données.
- Il récupère et nettoie les données du formulaire (`nom_produit` et `famille`).
- Si l'un de ces champs est vide, l'exécution s'arrête avec un message d'erreur.

#### 2. Vérification de la commercialisation :

- Le script vérifie si la gamme associée à la `famille` est toujours commercialisée en consultant le champ `commercialise` dans la table `products`.
- Si la gamme n'est plus commercialisée (`commercialise` vaut 0), un message d'erreur est affiché et l'exécution s'arrête.

#### 3. Détermination du `code_type` :

- Le script cherche si un produit avec le même `nom_produit` existe déjà dans la base.
- Si c'est le cas, il utilise le `code_type` existant.
- Sinon, il génère un nouveau `code_type` en prenant le maximum des `code_type` existants et en y ajoutant 1 (ou en commençant à 1 s'il n'y a aucun produit).

#### 4. Insertion du produit :

- Le produit est inséré dans la base avec les valeurs récupérées et calculées (`nom_produit`, `famille`, `code_type`) et avec `commercialise` défini à 1.

- Un message de succès est affiché indiquant que le produit a été ajouté ainsi que son code\_type.

#### 5. Gestion des erreurs et des méthodes non autorisées :

- En cas d'exception PDO, le script affiche un message d'erreur.
- Si la requête n'est pas de type POST, un message indiquant que la méthode n'est pas autorisée est affiché.

## **ANNEXE 2 : 1ere version de vue.php**

### **Liste des Produits**

Sélectionner une Famille :  Sélectionner un Nom du Produit :

Nom du Produit	Famille	Code Type	Barcode	Date d'Enregistrement	Commercialisation
FP8-RS/24V	FP8	1015	1	2025-02-17 09:50:19	Oui

### **Gérer la Commercialisation**

Code Type de la Gamme :

Statut :

## Annexe 3 : Produit.php – version 2



## Annexe 4 : explication de proces commande.php :

Ce code PHP gère la soumission d'un bon de commande via un formulaire en méthode POST. Voici les étapes principales :

1. **Vérification des données** : Il récupère les informations du formulaire (nom de la commande, destinataire, état et produits) et vérifie qu'elles sont valides (champ non vide, état valide, etc.).
2. **Transaction SQL** : Une transaction est démarrée pour garantir que toutes les modifications de la base de données (ajout de produits dans le bon de commande et mise à jour des stocks) soient effectuées ensemble ou pas du tout.
3. **Traitement des produits** : Pour chaque produit demandé, le code vérifie si suffisamment de stock est disponible. Si c'est le cas, le produit est ajouté au bon de commande et le stock est mis à jour.
4. **Gestion des erreurs** : Si un produit manque ou s'il y a un problème, une exception est levée et la transaction est annulée. Un message d'erreur est affiché.

5. **Confirmation** : Si tout est correct, la transaction est validée, un message de succès est affiché et l'utilisateur est redirigé vers une autre page.

En résumé, ce script crée un bon de commande, met à jour les stocks et gère les erreurs avec des messages de confirmation ou d'échec.


### Annexe 5 : Page code barra.php qui récupère les produits enregistré et génère les code-barres :

[Retour enregistrement produit](#)

### Liste des Produits avec leurs Codes-Barres

Filtrer par date :  [Réinitialiser la date](#)

Total des produits : 1

Nom du Produit	Code Type	Code-Barres	Fusion (Code Type + Barcode)	Étiquette
Tic Light-230V	1017	17-00001000		<a href="#">Afficher l'Étiquette</a>

### Annexe 6: Récupération des produits et explication de comment est généré l'image :

La liste des produits concernés est extraite de la base de données à l'aide d'une requête SQL :

```
$query = "SELECT id, nom_produit, barcode, code_type, created_at FROM products WHERE DATE(created_at) = :date ORDER BY id DESC LIMIT :limit OFFSET :offset";
```

### Explication de la requête :

- **SELECT id, nom\_produit, barcode, code\_type, created\_at** : sélectionne les informations essentielles du produit.
- **WHERE DATE(created\_at) = :date** : filtre les produits ajoutés à une date donnée (*paramètre dynamique*).
- **ORDER BY id DESC** : trie les résultats du plus récent au plus ancien.

- **LIMIT :limit OFFSET :offset** : permet la pagination des résultats.

Les produits récupérés sont ensuite stockés dans une variable \$products, qui sera utilisée pour générer dynamiquement le tableau HTML.

## Explication du processus de génération d'image des codes-barres :

Le script de génération des codes-barres repose sur la bibliothèque **JsBarcode**, un outil JavaScript permettant de créer des codes-barres en 1D directement sur un élément <canvas> ou <svg> dans une page web.

## Chargement de la bibliothèque JsBarcode :

Avant de pouvoir générer les codes-barres, la bibliothèque est chargée à l'aide du script externe suivant :

```
<script
src="https://cdn.jsdelivr.net/npm/jsbarcode@3.11.0/dist/JsBarcode.all.min.js">
</script>
```

## Explication de l'image générée :

L'image générée est un **fichier PNG** contenant **deux codes-barres distincts fusionnés** sur un seul élément graphique. Elle est constituée des éléments suivants :

### 1. Le code-barres du type de produit (code\_type)

- Il est placé en haut de l'image.
- Il représente une **référence unique** à la gamme du produit.
- Son affichage inclut une version textuelle de la valeur en dessous des barres.

### 2. Le code-barres du numéro de série (barcode)

- Il est positionné juste en dessous du premier code-barres.
- Il correspond à un **identifiant unique** propre à chaque produit.
- Il est également accompagné de sa valeur sous forme de texte.

### 3. Mise en page et alignement

- L'image a une **largeur de 600 px** et une **hauteur de 120 px**.
- Les deux codes-barres sont **centrés horizontalement** pour une meilleure lisibilité.
- L'espacement entre les codes-barres est ajusté pour éviter toute confusion lors du scan.

### 4. Format et stockage

- L'image est **générée dynamiquement en mémoire** sur un <canvas>.
- Elle est **convertie au format PNG** via `canvas.toDataURL( 'image/png' )`.

- Elle est ensuite **envoyée au serveur** sous forme de **donnée base64** pour être sauvegardée et réutilisée ultérieurement.

Ce format d'étiquette permet une **identification rapide et fiable** du produit, facilitant ainsi son suivi et sa traçabilité dans le système de gestion. 🚀

## Annexe 7: Page historique commande.php :

### Historique des commandes

Nom de la commande	Destinataire(s)	Date (Complétude ou Création)	Quantité de produits	Détails	État de la commande
F-PO-2501-2075	BTIB	2025-01-24 10:32:32	1	<a href="#">Voir les détails</a>	totale

### Résumé du code : Historique des commandes

Le fichier **historique\_commandes.php** permet d'afficher l'historique des commandes passées par l'utilisateur dans une interface web. Le système récupère les commandes de la base de données et les présente sous forme de tableau, incluant des informations comme le nom de la commande, les destinataires, la date de création ou de complétion, le nombre de produits, et l'état de la commande.

### Fonctionnalités principales :

- Affichage des commandes passées, triées par date de création ou de complétion.
- Chaque commande inclut un lien pour afficher ses détails.
- Un bouton permet à l'utilisateur d'enregistrer une nouvelle commande.

### Requête SQL :

La requête SQL suivante récupère les informations relatives aux commandes :

```
SELECT
    b.nom_commande,
    GROUP_CONCAT(DISTINCT d.nom_destinataire SEPARATOR ', ') AS destinataires,
    MAX(COALESCE(b.date_complétude, b.date_creation)) AS date_affichage,
    COUNT(*) AS quantite_totale,
    b.etat_commande
FROM
    bon_de_commande AS b
LEFT JOIN
    destinataire AS d ON b.destinataire = d.nom_destinataire
GROUP BY
    b.nom_commande, b.etat_commande
ORDER BY
    date_affichage ASC
```



**Explication de la requête :**

- La requête sélectionne le nom de la commande (nom\_commande), les destinataires associés (GROUP\_CONCAT pour concaténer plusieurs destinataires), la date de complétion ou de création (MAX(COALESCE(...))), le nombre total de produits associés à la commande (COUNT(\*)), et l'état de la commande (etat\_commande).
- Les données sont récupérées depuis les tables **bon\_de\_commande** et **destinataire**, avec une jointure **LEFT JOIN** pour associer les destinataires à chaque commande.
- Les résultats sont groupés par nom de commande et état de la commande, et triés par la date d'affichage (date\_affichage), qui est la date de complétion ou de création de la commande.

**Annexe 8: Exemple d'un détail d'une commande :**

**Détails de la commande : F-PO-2501-2075**

Destinataire : BTIB

Date d'enregistrement : 2025-01-24 10:32:32

Quantité de produits : 1

État de la commande : totale

Passer à l'état Totale

Retour à l'historique

**Résumé des Quantités par Produit**

Nom Produit	Quantité Totale
Tic Light-230V	1

**Produits dans la commande**

Code Type: 1017 | Produit : Tic Light-230V ▼

Barcode	Famille	Nom Produit	Date d'enregistrement
17-00001000	Tic Light	Tic Light-230V	2025-01-24 10:32:32

**Objectif :** Cette page PHP permet de récupérer et d'afficher les détails d'une commande, y compris ses produits, ainsi que de modifier l'état de la commande.

### **1. Initialisation et Sécurité :**

- Vérification de la présence d'un `commande_id` dans l'URL.
- Démarrage de la session et vérification de l'authentification de l'utilisateur.

### **2. Requêtes SQL :**

- Récupération des informations générales sur la commande (nom, destinataire, dates, état).
- Récupération des détails des produits associés à la commande et du nombre d'articles par produit.

### **3. Modification de l'état de la commande :**

- L'utilisateur peut changer l'état de la commande entre "Totale" et "Partielle".
- Mise à jour de l'état dans la base de données et rafraîchissement de la page.

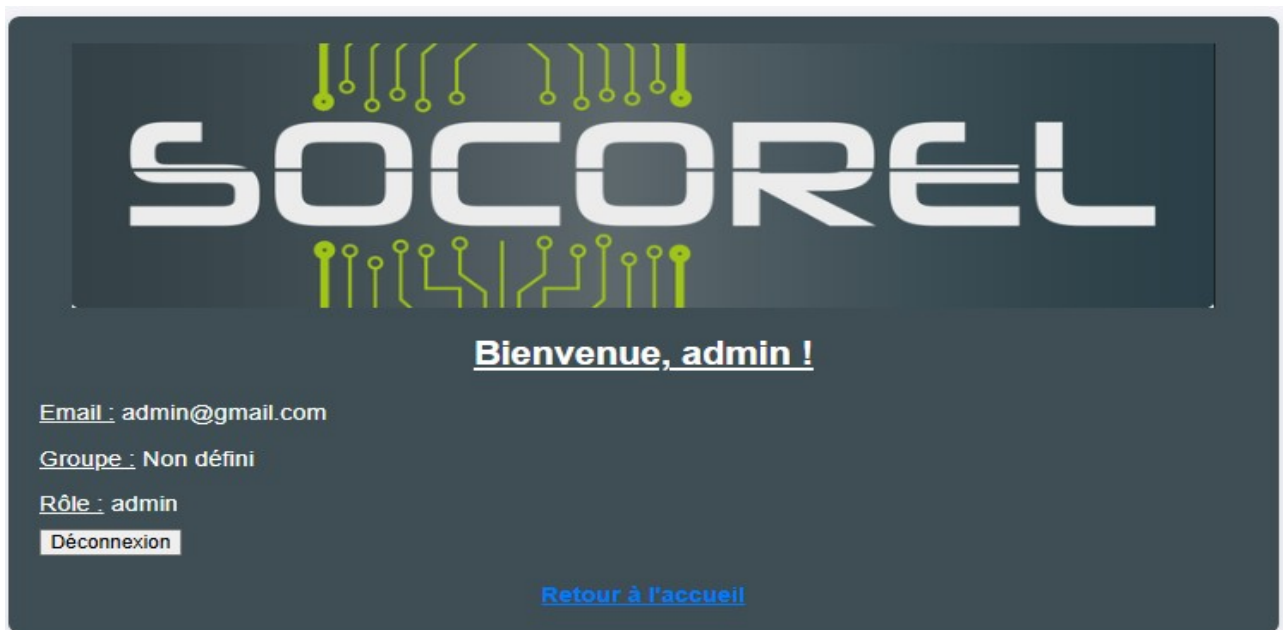
### **4. Affichage des informations :**

- Affichage des informations de la commande (nom, destinataire, dates).
- Tableau récapitulatif des produits avec le nombre total d'articles par produit.
- Possibilité de passer à la page de complétion de la commande si l'état est "Partielle".

### **5. Interface utilisateur :**

- Un script JavaScript permet de masquer/afficher les détails des produits par catégorie.
- Des boutons permettent de revenir à l'historique des commandes ou de modifier l'état de la commande.

### **Annexe 9: Zone de connexion (page : Login.php) :**



### **Annexe 10: Accès utilisateurs en fonction de son rôle :**

Le code présenté permet d'afficher des boutons de navigation dynamiques sur une interface web, en fonction du rôle et du groupe de l'utilisateur connecté. Voici le fonctionnement détaillé :

#### **1. Vérification du rôle de l'utilisateur :**

- Le code vérifie si la variable de session `role` est définie, ce qui indique que l'utilisateur est connecté.

#### **2. Affichage des options pour les administrateurs :**

- Si l'utilisateur a le rôle "admin", plusieurs boutons sont affichés pour lui permettre d'accéder à des pages spécifiques comme la gestion des utilisateurs, l'enregistrement de produits, l'historique des commandes, etc.

#### **3. Affichage des options pour les utilisateurs du groupe "declic" :**

- Si l'utilisateur est dans le groupe "declic", mais avec un rôle "utilisateur", seulement certaines pages sont accessibles, telles que l'enregistrement de produits et la consultation de l'historique des commandes et produits.

#### **4. Affichage des options pour les utilisateurs du groupe "btib" :**

- Si l'utilisateur appartient au groupe "btib", il aura accès à des pages spécifiques pour consulter des informations sur les produits et gérer la réception des commandes.

#### **5. Redirection vers les pages :**

- Chaque bouton, selon l'option affichée, redirige l'utilisateur vers une page spécifique via l'attribut `onclick` qui modifie la localisation de la page (location.href) pour la nouvelle destination.



## Annexe 11: gestion de la commercialisation – version 2 :

Gestion des Produits				
Famille	Code Type	Nom Produit	Commercialisation	Action
R TIC	1028	R TIC	Commercialisé	Décommercialiser
Tic Light	1017	Tic Light-230V	Commercialisé	Décommercialiser

[Retour à Gestion des produits](#)

## Annexe 12: Modification des champs nom commande et destinataire :

**Modifier les détails de la commande**

Nouveau nom de la commande :

Nouveau destinataire :

[Retirer des produits](#)
[Enregistrer](#)
[Annuler](#)

### Annexe 13: page software hardware.php pour gérer les différentes version de chaque produit :

Ici nous avons comme exemple le Produit : Tic Light-230V avec ces deux versions, les deux hardware, les deux software et la possibilité avec la dernière version actuel, d'ajouter une nouvelle version qui reprend les champs de la version actuel et qui laisse la possibilité à l'administrateur, la possibilité de changer les champs software et hardware, puis confirmer cette nouvelle valeur et l'enregistrer dans la base de données.

#### Ajouter une nouvelle version pour Tic Light-230V

Code Type :	<input type="text" value="1017"/>	Nom Produit :	<input type="text" value="Tic Light-230V"/>	Famille :	<input type="text" value="TIC Light"/>	Hardware :	<input type="text" value="4.5"/>
		Software :	<input type="text" value="6.3"/>	Version :	<input type="text" value="3"/>	Date de mise à jour :	<input type="text" value="18/02/2025 11:26"/>
				<input type="button" value="Ajouter"/>			

1.4	2.1	TIC Light	Non modifié	1	
-----	-----	-----------	-------------	---	--

### Annexe 14: formulaire enregistrement de produit – version 3 :

### Enregistrer un Nouveau Produit

**Famille :**

**Nom du Produit :**

Version	Hardware	Software	Choisir
1	1.4	2.1	<input type="radio"/>
2	8.6	7.6	<input type="radio"/>


**Quantité à Ajouter :**

ajout d'une zone pour sélectionner la version du produit à enregistrer dans la base de données.

### Annexe 15: affiche d'un produit dans la page vue.php – version 2 :

Famille	Nom du Produit	Code Produit	Barcode	Date d'Entrée stockage	Enregistrement bon de commande	Stockage	Référence Commande	Destinataire	Hardware	Software	Commercialisation
Tic Light	Tic Light-230V	1017	17-00001000	2025-01-23 15:45:13	2025-01-24 10:32:32	Envoyé	F-PO-2501-2075	BTIB	8.6	2.1	Oui

## Annexe 16: Page gestion des utilisateurs + création d'un nouvel utilisateurs :



Info Utilisateur

## Gestion des utilisateurs

### Liste des utilisateurs

ID	Nom	Email	Groupe	Rôle	Actions
1	admin	admin@gmail.com	Aucun groupe	admin	<input type="text" value="Nouveau groupe"/> <input type="button" value="Modifier le groupe"/> <input type="button" value="Supprimer"/>

### Créer un nouvel utilisateur

Nom : 
 Email : 
 Mot de passe : 
 Rôle : 
 Groupe :

## Annexe 17: Scan d'un code type représentant une gamme :




Famille: Tic Light



Info Utilisateur



## Scanner un Code-Barres

Entrez le code-barres :

### Produits associés

Nom	Famille	Numéro de série	Version	Hardware	Software	Date enregistrement du produit
Tic Light-230V	Tic Light	17-00001000	1	1.4	2.1	2025-01-23 15:45:13

**Annexe 18: Scan d'un barcode représentant le numéro de série d'un produit :**



Famille: Tic Light

Scanner un Code-Barres

Entrez le code-barres :

Tic Light-230V (17-00001000)

Famille	Tic Light
Numéro de série	17-00001000
Version	1
Hardware	1.4
Software	2.1
date enregistrement du produit	2025-01-23 15:45:13