# BTS SIO 2A 2024/2025 - SLAM

ProjetSpé 2 : Deliver’easy

## Contexte

Une entreprise de livraison nommée **Deliver’easy** vous contacte afin de développer une application web de gestion et d’organisation de tournées de livraison de colis en France Métropolitaine. Cette application a pour but (dans un premier temps) d’être utilisée de 2 manières :

- **Au siège de l’entreprise** : sur leur **poste de travail**, afin de pouvoir **concevoir et mettre à jour facilement les tournées de livraison** des livreurs de l’entreprise, il leur faut pouvoir **saisir les clients, les livreurs, les adresses, les commandes et les colis**.
- **En extérieur** : sur **mobile**, les livreurs veulent pouvoir accéder à l’application pour **consulter les informations sur leur tournée en cours** (infos sur l’étape en cours (client / adresse / coordonnées de contact), et **pointer les livraisons faites** (noter l’heure de livraison, pouvoir faire signer le client, marquer un commentaire). En cas d’impossibilité de livrer les colis d’une commande (client indisponible, mauvaise adresse, etc…), les livreurs veulent pouvoir **signaler une commande à replanifier** par le personnel au siège de l’entreprise.

L'application doit permettre une **gestion efficace des adresses**, **des livreurs** de l’entreprise, **des clients**, **des colis** transportés, **des commandes** passées par les clients pour livrer les colis et des livraisons, ainsi qu'une planification des **tournées quotidiennes** en **associant un livreur, une date de tournée, et un ensemble de livraisons** planifiées.

*L’optimisation automatique de la tournée (le chemin le plus court) d’un livreur n’est pas à prendre en compte dans ce sujet, mais peut être abordée si vous finissez le projet plus vite que prévu.*

## Consignes Techniques

### Backend

- **Développer une API RESTful** pour gérer la communication entre le frontend et la base de données(avec Node.js et Express par exemple, si votre précédent projet fonctionne et que d’autres technologies vous intéressent, me demander).
- Configurer un environnement sécurisé pour **l’authentification des utilisateurs** et les **contrôles d'accès**.
- Organiser le code en utilisant une **structure modulaire** pour **faciliter la maintenance et l’extension de l'application** (contrôleurs, routes, middlewares).
- Implémenter des **validations robustes côté serveur pour les entrées utilisateurs** afin de garantir la **cohérence des données**.
- Utiliser des outils comme **JWT** pour **l'authentification et la gestion des sessions utilisateur**.
- Assurer la **gestion des erreurs et des logs** pour **le suivi et la résolution des problèmes** potentiels.

# Frontend

- Créer une **interface utilisateur moderne et réactive** (avec Vue.js 3 et en utilisant l'API de Composition, si votre précédent projet fonctionne et que d'autres technologies vous intéressent, me demander), en tirant parti du framework utilisé pour une organisation optimale du code (pas de redondance).
- Développer des composants réutilisables pour les formulaires de saisie, les tableaux, et les éléments de navigation.
- Assurer une navigation fluide entre les différentes sections grâce à un routeur frontend, avec des routes sécurisées selon les rôles des utilisateurs.
- Implémenter des fonctionnalités de recherche et de filtres avancés pour naviguer facilement dans les données (par exemple, recherche de livraisons par date ou par livreur).
- Utiliser des librairies CSS pour un design épuré et responsive (adaptative au support de navigation utilisé), garantissant une expérience utilisateur optimale sur différents types de périphériques (desktop, tablette, mobile).
- **Base de Données**
    - Concevoir et structurer une **base de données relationnelle** en suivant les principes de normalisation pour **éviter les redondances** et **améliorer l'efficacité**.
    - A l'aide de la **méthode Merise**, définir le **dictionnaire de données**, les dépendances fonctionnelles, ainsi que le MCD détaillant les entités et leurs associations.
    - Écrire des **scripts SQL** pour la **création du schéma relationnel** ainsi que pour **l'insertion de données de test**, avec des **contraintes appropriées** (clés primaires, clés étrangères, contraintes d'intégrité).

# Fonctionnalités Fonctionnelles

- **Gestion des Entités** :
    - CRUD (Create, Read, Update, Delete) complet pour toutes les entités avec des validations et contraintes appropriées.
    - Fonctionnalités de recherche et de filtrage par différents critères (nom, date, référence, etc.).
- **Planification des Livraisons** :
    - Saisie des heures de livraison prévues pour organiser les tournées quotidiennes.
    - Association d'un livreur, d'une date et d'un ensemble de livraisons pour former une tournée.
    - Interface permettant de visualiser les tournées par date et par livreur avec un résumé des livraisons planifiées (par exemple, nombre de colis, poids total, adresses de livraison).
- **Sécurité & Gestion des Comptes** :
    - Authentification des utilisateurs et gestion des rôles (administrateurs, livreurs).
    - Gestion des permissions pour garantir que les utilisateurs n'ont accès qu'aux fonctionnalités qui leur sont autorisées.

# Livrables

Sont attendus **à minima** en sortie de ce projet pour être évalué :

1. **MCD (Modèle Conceptuel de Données)** :
   Un schéma détaillant les entités, leurs attributs et les relations entre elles pour une vue d'ensemble de la base de données.
2. **Script SQL d'Initialisation de la Base de Données** :
   Un script SQL complet pour créer les tables, contraintes, et données initiales nécessaires à l'application.
3. **Code Source du Backend et du Frontend** :
   Le code complet et documenté des API ainsi que du frontend de votre projet.
4. **Schéma d'Architecture Applicative** :
   Un schéma simple illustrant l'architecture de la solution développée, incluant les flux d'échange de données entre le frontend, le backend, et la base de données.