# Dossier de Soutenance - Projet Elyne

## Introduction

Le Projet Elyne est destiné à organiser le transport de marchandises. Initialement, le module avait été conçu pour l'enlèvement de déchets d'un chantier A à un chantier B. Lors de la prise en main du projet, l'utilisateur principal du module Elyne, intégré à Dolibarr, était le transporteur.

Une évolution est envisagée afin que l'utilisateur du module devienne le donneur d'ordre, c'est-à-dire le tiers demandant un transport.

## I. Diagramme d'activité

Un diagramme d'activité a été présenté à Antoine Nguyen afin de vérifier la compréhension de la mécanique globale du module. Certaines étapes ont été simplifiées.

Suite à ce diagramme, nous avons décidé que :

* Un lieu est toujours un projet.
* Le lieu d’enlèvement correspond à un projet distinct du lieu de destination.
* Il est donc nécessaire de créer deux projets par demande de transport.
* Chaque projet est relié à un tiers, qui peut être le même ou un tiers différent.

### Exemple :

J.G Maçonnerie demande l’enlèvement de marchandise en lieu A vers un lieu B.

* Projet A : Tiers = J.G Maçonnerie, Lieu = J.G Maçonnerie Laval.
* Projet B : Tiers = Déchetteries de France, Lieu = Déchetterie de Laval.

## II. Simplification du module

Un second diagramme a été établi pour visualiser la simplification du processus.

## III. Réunion client et réorganisation du développement

Suite à la réunion avec le client, les délais ont été clarifiés.  
Nous avons décidé de répartir le projet entre deux développeurs, avec un délai de 2 mois pour finaliser l’application.

Ma mission consiste dans les deux premières semaines à préparer l’application tablette destinée aux chauffeurs.

### 1. Maquette de l’application

J’ai conçu la maquette afin de visualiser les composants nécessaires. Le workflow d’une lettre de voiture comprend 4 étapes principales :

1. Commande existante mais non assignée à un chauffeur.
2. Assignée à un chauffeur.
3. Marchandise chargée.
4. Marchandise en transport.
5. Marchandise déchargée, commande clôturée.

Certains composants de la lettre de voiture sont récurrents, parfois éditables et parfois en lecture seule. Il est donc nécessaire de créer des composants modulables selon le contexte.

De plus, un besoin d’héritage se fait ressentir, car le destinataire et l’expéditeur de la marchandise auront exactement les mêmes interventions :

* Inscription des coordonnées.
* Signature pour officialiser la lettre de voiture.

Ce document tripartite implique trois intervenants ayant des rôles différents mais agissant ensemble pour produire un document final unique.

### 2- **Établissement d’un diagramme de classe et définition de l’architecture du projet**

Une fois les composants schématisés et les éléments qu’ils doivent contenir définis, j’ai créé un diagramme de classe afin de mettre en place une architecture orientée composants, comprenant les couches suivantes : Model, Vue, Repository, Controller et Composants. Le dossier Composants fait appel aux repositories pour générer le template final nécessaire à l'affichage.

* Model (Entités) : Représente les objets métier et la structure des données.
* Vue (Pages HTML injectant les composants) : Gère l'affichage des informations en utilisant les composants générés.
* Composants (Construction du template final) :
  + Lists : Permet de boucler et d'afficher des cartes ou éléments sous forme de liste.
  + Card : Représente une carte ou un élément individuel dans la liste.
  + Boutons : Gère les boutons et interactions utilisateurs.
  + Header et Footer : Gère l'entête et le pied de page des templates.
* Controllers (Utilisation du repository) : Prépare et passe les données aux vues en interagissant avec les repositories.
* Repository (Requêtes SQL utilisant les accesseurs des modèles) : Gère les accès aux données, interroge la base de données et retourne les résultats nécessaires aux controllers.

La nécessité d’utiliser des composants modulaires (comme en React) et la volonté de bien structurer le code m’ont conduit à choisir cette architecture. Travaillant dans un environnement Dolibarr, qui limite l'implémentation de frameworks et de bibliothèques modernes, j’ai opté pour cette approche afin de faciliter la prise en main par le développeur qui continuera le projet une fois ma formation terminée.

L’implémentation du diagramme de classe a mené à quelques **questions** notamment la désignation des classes pour les éléments constitutifs de la lettre de voiture. En effet ces différents éléments sont visibles dans l’application initiale parfois sous le nom de « commande » , parfois sous le nom de « missions » pour le point de vue chauffeur et parfois sous le nom de « chantier ».

Cela a également mené à refléxion sur la relation entre la « mission » et l’entité « location » qui est nécessaire tant pour le lieu de chargement que le lieu de déchargement.