Documentation technique

Pro à Pro - Vision des commandes sur une carte - 2024

Table des matières

I. Introduction

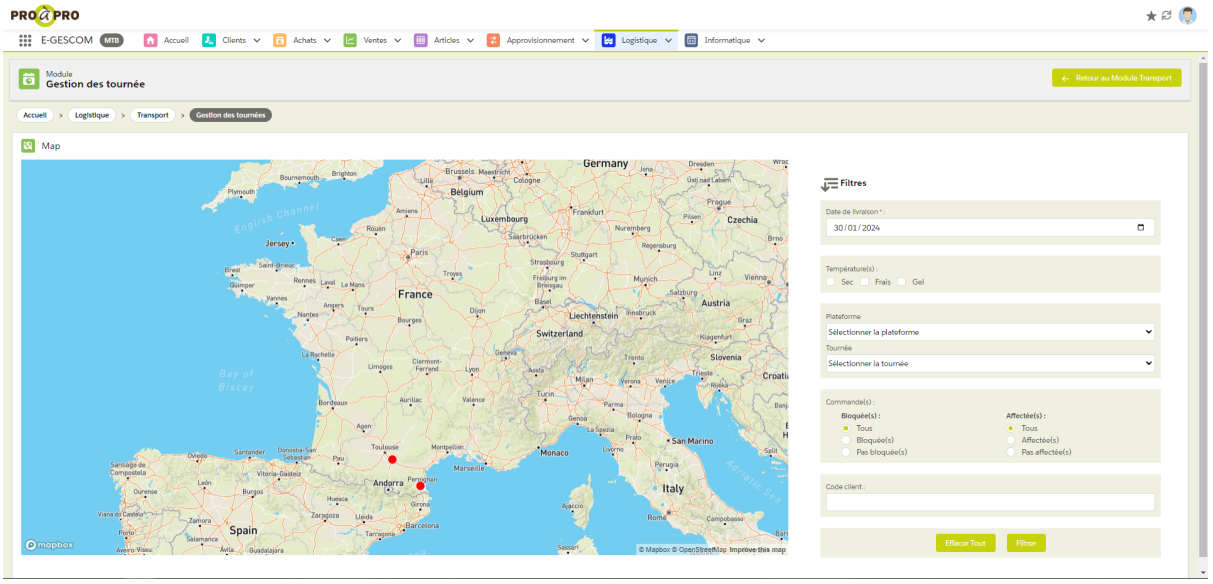
A. Entreprise

Pro à Pro est une entreprise française d'environ 1 000 salariés. Elle est spécialisée dans la distribution de produits alimentaires frais et de qualité. Avec son réseau de logistique avancé, Pro à Pro livre ses clients en 24h.

B. Objectif

L'objectif de la fonctionnalité est de pouvoir visualiser les commandes clients sur une carte. Cela permet de différencier les clients et savoir si leurs commandes sont affectées à une tournée, ou si elle est bloquée. La fonctionnalité permet aussi de trier par tournée, cela permet d'avoir toutes les informations sur celle-ci (Chauffeur, Camion, température, poids, volume, etc). Sur une commande, si le bon de préparation est édité, il est possible de le visualiser pour avoir le détail de la commande.

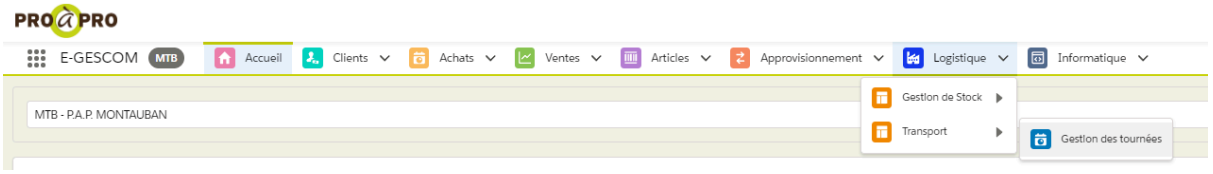
C. Vue d'ensemble



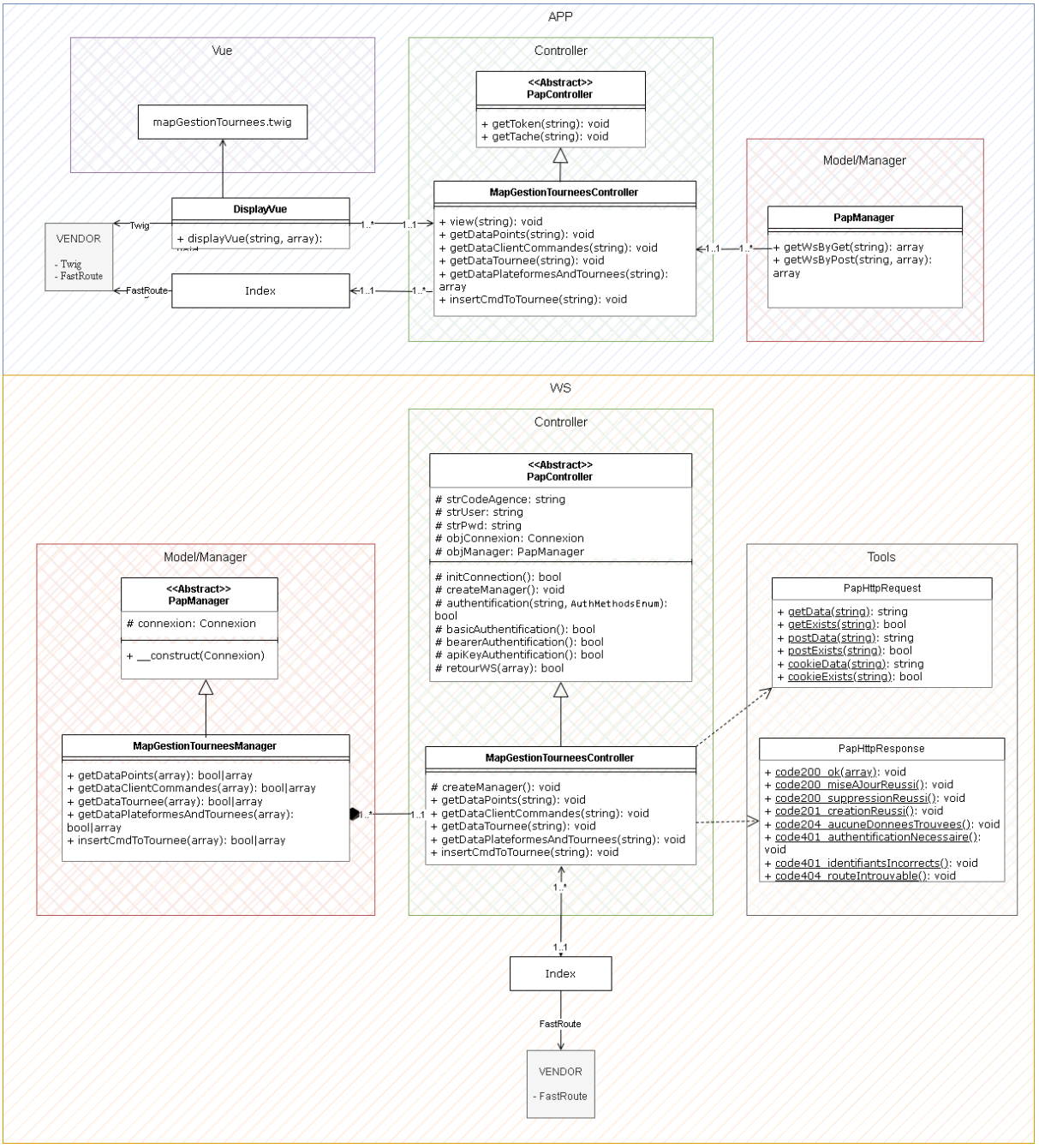
II. Architecture Global

A. Positionnement de la fonctionnalité

La fonctionnalité se trouve dans l'application web E-Gescom. Dans la catégorie "Logistique > Transport > **Gestion des tournées**". On peut y accéder grâce à la barre de navigation ou par la page d'accueil. Pour pouvoir accéder à cette fonctionnalité il faut avoir la tâche "GCATPI".

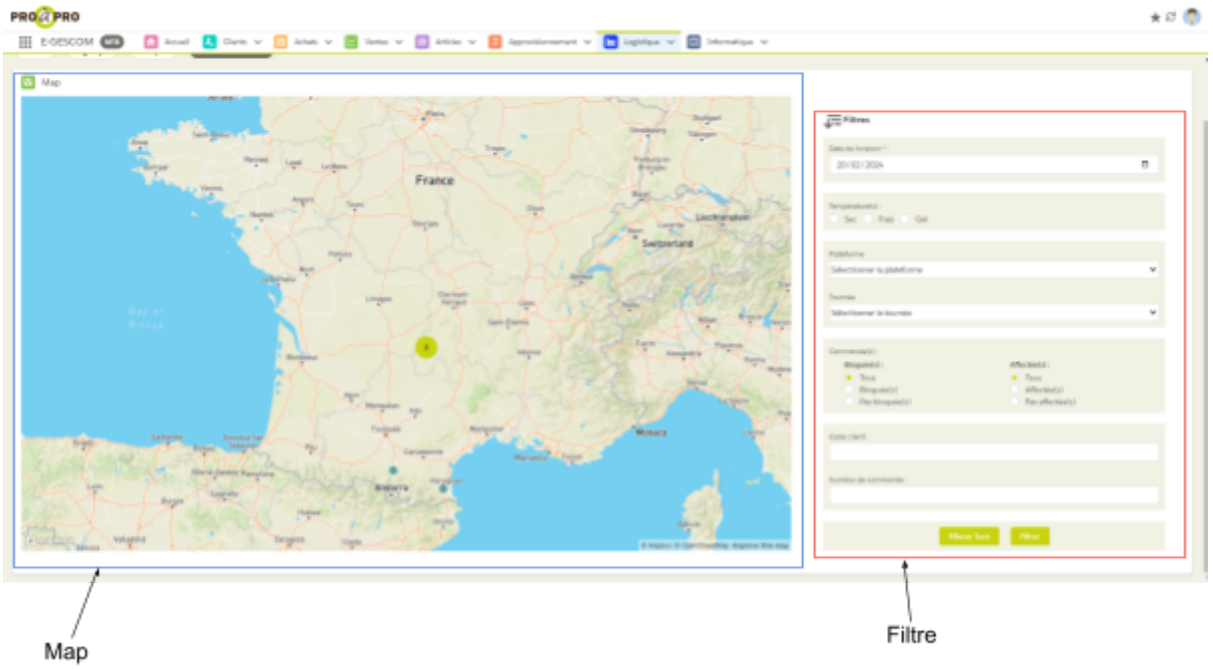


B. UML


APP
Vue
mapGestionTournees.twig
DisplayVue
+ displayVue(string, array): void
VENDOR
- Twig
- FastRoute
Twig
FastRoute
Index
Controller
<<Abstract>>
PapController
+ getToken(string): void
+ getTache(string): void
MapGestionTourneesController
+ view(string): void
+ getDataPoints(string): void
+ getDataClientCommandes(string): void
+ getDataTournee(string): void
+ getDataPlateformesAndTournees(string): array
+ insertCmdToTournee(string): void
Model/Manager
PapManager
+ getWsByGet(string): array
+ getWsByPost(string, array): array
1..1
1..*
WS
Controller
<<Abstract>>
PapController
strCodeAgence: string
strUser: string
strPwd: string
objConnexion: Connexion
objManager: PapManager
initConnection(): bool
createManager(): void
authentification(string, AuthMethodsEnum): bool
basicAuthentification(): bool
bearerAuthentification(): bool
apiKeyAuthentification(): bool
retourWS(array): bool
Model/Manager
<<Abstract>>
PapManager
connexion: Connexion
+ __construct(Connexion)
MapGestionTourneesManager
+ getDataPoints(array): bool|array
+ getDataClientCommandes(array): bool|array
+ getDataTournee(array): bool|array
+ getDataPlateformesAndTournees(array): bool|array
+ insertCmdToTournee(array): bool|array
MapGestionTourneesController
createManager(): void
+ getDataPoints(string): void
+ getDataClientCommandes(string): void
+ getDataTournee(string): void
+ getDataPlateformesAndTournees(string): void
+ insertCmdToTournee(string): void
Tools
PapHttpRequest
+ getData(string): string
+ getExists(string): bool
+ postData(string): string
+ postExists(string): bool
+ cookieData(string): string
+ cookieExists(string): bool
PapHttpResponse
+ code200_ok(array): void
+ code200_miseAJourReussi(): void
+ code200_suppressionReussi(): void
+ code201_creationReussi(): void
+ code204_aucuneDonneesTrouvees(): void
+ code401_authentificationNecessaire(): void
+ code401_identifiantsIncorrects(): void
+ code404_routeIntrouvable(): void
1..*
1..1
Index
FastRoute
VENDOR
- FastRoute

III. Description de l'interface utilisateur

L'interface utilisateur sur E-GESCOM pour la fonctionnalité de vision des commandes sur une carte se présente en deux parties. La première est la map avec un zoom sur la France. La seconde partie est le filtre.



A. Filtre

Le filtre contient 8 champs :

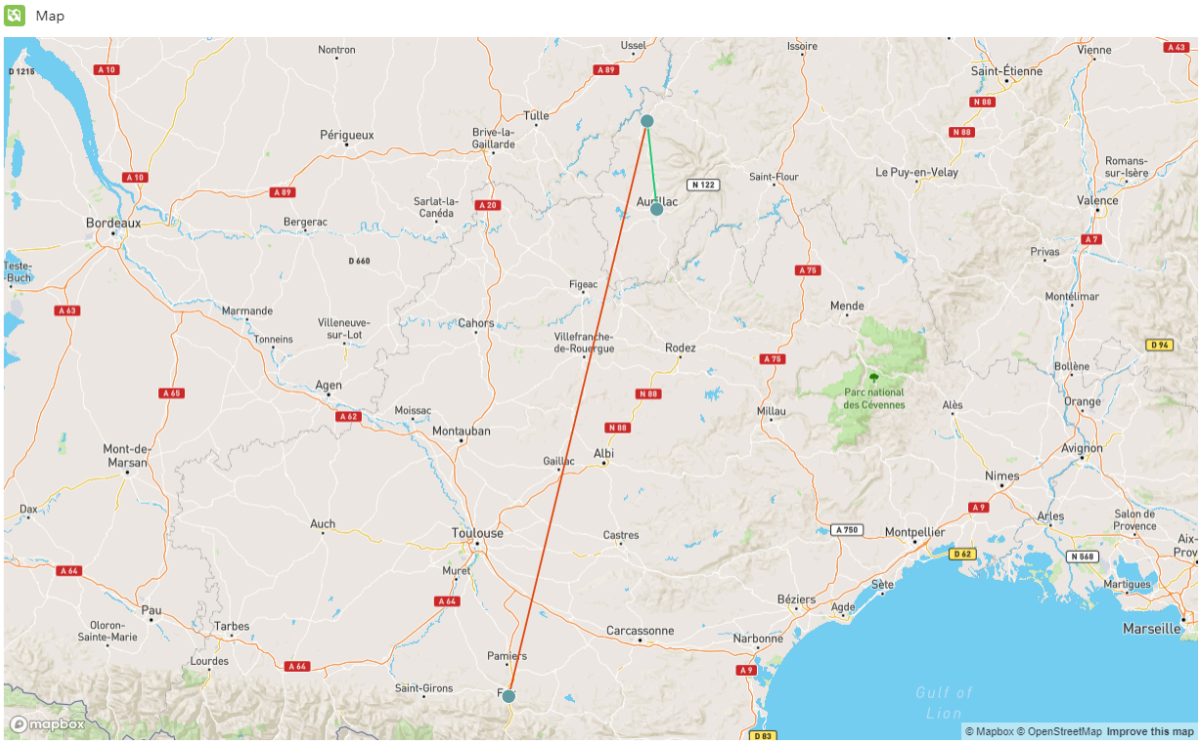
- **Date de livraison** : *Permet de filtrer sur la date de livraison d'une commande. (Obligatoire)*
- **Températures** : *Permet de filtrer sur la température des commandes (Sec, Frais ou Gel).*
- **Plateforme** : *Permet de filtrer sur la plateforme de départ de la tournée de livraison.*
- **Tournée** : *Permet de filtrer sur la tournée de livraison.*
- **Commande bloquée** : *Permet de filtrer sur les commandes bloquées ou non bloquées.*
- **Commande affectée** : *Permet de filtrer sur les commandes affectées ou non affectées.*
- **Code client** : *Permet de filtrer sur le code client.*
- **Numéro commande** : *Permet de filtrer sur le numéro de la commande.*

Le filtre contient aussi 2 boutons :

- **Effacer tout** : *Permet d'effacer tous les éléments du filtre et le résultat afficher sur la map.*
- **Filtrer** : *Permet de filtrer avec les champs remplis dans le filtre.*

B. Map

La map permet d'afficher la résultante du filtre après avoir appuyé sur le bouton Filtrer. Elle affichera tous les clients qui ont une commande qui doit être livrée sur la date de livraison et remplissant les critères du filtre.
Les différents points seront reliés en fonction des tournées des commandes.



Si on clique sur un point on obtient les informations du client, puis les informations de chaque commande du client pour cette date de livraison.

Infos livraison

Client : VVF Villages
Adresse : Vendes 15240 BASSIGNAC
Nombre de commande : 2
Commande bloquée : 2
Commande affectée : 2

T°C de transport	Ambiante (20°)	Dirigée (5°)	Surgelée (-18°)	Total
Volume (m^3)	3.180	0.486	0.000	3.666
Poids (kg)	1120.94	165.40	0.00	1286.34
Nombre de lignes	4	1	0	5
Equivalent palette	3.58	0.92	0.00	4.50

Commandes :

Cde 1686605 Cde 1686606

Commande : 1686605
Plateforme : 640
Tournée : 64
Bloquée : Oui
Affectée : Oui

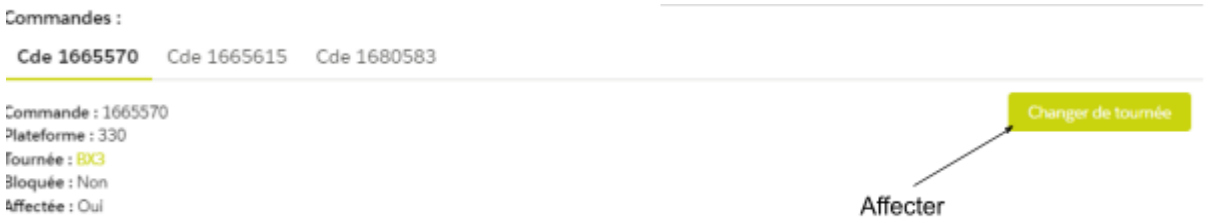
Changer de tournée

T°C de transport	Ambiante (20°)	Dirigée (5°)	Surgelée (-18°)	Total
Volume (m^3)	0.570	0.486	0.000	1.056
Poids (kg)	128.12	165.40	0.00	293.52
Nombre de lignes	2	1	0	3
Equivalent palette	1.08	0.92	0.00	2
Bon de préparation	Aucun	Aucun	Aucun	

En premier lieu nous avons les informations du client avec son nom, adresse et son nombre de commandes. Ensuite Il y a un tableau résumant le volume, poids, lignes et palettes par températures de toutes les commandes. Enfin il y a des onglets par commande permettant d'avoir les informations sur celle-ci avec le numéro, la plateforme de départ, la tournée, si elle est bloquée, si elle est affectée et un tableau résumant les informations de la commande par température.

Si on filtre sur un code tournée et qu'il y a bien des commandes à livrer sur la date de livraison renseignée. On verra apparaître un petit camion résumant les informations de la tournée comme la plateforme de départ, code tournée, le transporteur, le nom du chauffeur, le camion ou bien la température.

Il est possible d'affecter ou de changer l'affectation d'une commande à une tournée en cliquant sur le bouton se trouvant dans l'onglet d'une commande. Il suffit juste de rentrer les informations demandées pour affecter celle-ci.



IV. Backend (Côté Serveur)

A. Utilisation de fonction

1. Vérification du token

- **Nom** : *checkToken*
- **Description** : *Vérifier si l'utilisateur est bien connecté avant de lui afficher la page.*
- **Classe** : *PapController*
- **Paramètre** : *Aucun*
- **Retour** : *Aucun*
- **Exemple** : `$this->checkToken();`
- **Code** :

```
public function checkToken(): void
{
    if (empty($_SESSION["token"])) {
        header("Location: /app/");
    }
}
```

2. Vérification de la tâche

- **Nom** : *checkTache*
- **Description** : *Vérifier si l'utilisateur à les droits pour pouvoir accéder à la page.*
- **Classe** : *PapController*
- **Paramètre** :
  - Nom de la tâche → Obligatoire
- **Retour** : *Aucun*
- **Exemple** : `$this->checkTache("GESWPLREC");`
- **Code** :

```
public function checkTache(string $tache): void
{
    $tache = strtoupper($tache);
    $agence = $_GET["agence"];
    if (array_key_exists($tache, $_SESSION["Tache_Menu_Dashboard"]) &&
($_SESSION["Tache_Menu_Dashboard"][$tache]["ACCES"] != "O")) {
        header("Location: /app/dashboard?agence=" . $agence);
        exit();
    }
}
```

B. Nouveaux Web Services

1. Points client

- **Description** : *Permet de récupérer tous les clients sur une date.*
- **Route** :
  - ➢ **API :** *POST*
  - ➢ **Paramètres** *:*
    - Date de livraison ⟶ Obligatoire
    - Température (SEC, FRAIS, GEL)
    - Plateforme de départ
    - Tournée
    - Commandes (Bloquée : OUI | NON | Tous, Affectée : OUI | NON | TOUS)
    - Code client
    - Numéro commande
  - ➢ **URL** :
    /ws/logistique/transport/gestion-des-tournees/{agence}/data
- **Code** :
  - ➢ **Contrôleur** : *GestionDesTourneesController*
  - ➢ **Manager** : *GestionDesTourneesManager*

2. Commandes par client

- **Description** : *Permet de récupérer toutes les commandes pour un client*
- **Route** :
  - ➢ **API** : *GET*
  - ➢ **Paramètre** :
    - Date de livraison ⟶ Obligatoire
  - ➢ **URL** :
    /ws/logistique/transport/gestion-des-tournees/{agence}/clients
- **Code** :
  - ➢ **Contrôleur** : *GestionDesTourneesController*
  - ➢ **Manager** : *GestionDesTourneesManager*

3. Tournée

- **Description** : *Permet de récupérer les informations sur une tournée sur une date.*
- **Route** :
  - ➢ **API** : *GET*
  - ➢ **Paramètres** :
    - Date de livraison ⟶ Obligatoire
    - Code tournée ⟶ Obligatoire
  - ➢ **URL** :
    /ws/logistique/transport/gestion-des-tournees/{agence}/tournee
- **Code** :
  - ➢ **Contrôleur** : *GestionDesTourneesController*
  - ➢ **Manager** : *GestionDesTourneesManager*

4. Plateformes et tournées

- **Description** : *Permet de récupérer les plateformes et tournées lier à une plateforme.*
- **Route** :
  - ➢ **API** : *GET*
  - ➢ **URL** : */ws/logistique/transport/gestion-des-tournees/{agence}/plateformes-and-tournees*
- **Code** :
  - ➢ **Contrôleur** : *GestionDesTourneesController*
  - ➢ **Manager** : *GestionDesTourneesManager*

5. Insérer une commande dans une tournée

- **Description** : *Permet d'insérer une commande client dans une tournée.*
- **Route** :
  - ➢ **API** : *GET*
  - ➢ **Paramètres** :
    - Agence → Obligatoire
    - Code commande → Obligatoire
    - Date de livraison → Obligatoire
    - Plateforme de départ → Obligatoire
    - Code tournée → Obligatoire
    - Position dans la tournée → Obligatoire
    - Numéro de BP (Bon de préparation) → Obligatoire
  - ➢ **URL** : */ws/logistique/transport/gestion-des-tournees/{agence}/insert-tournee*
- **Code** :
  - ➢ **Contrôleur** : *GestionDesTourneesController*
  - ➢ **Manager** : *GestionDesTourneesManager*

V. Frontend (Côté Client)

A. Langages et technologies

- **TWIG** : *Moteur de rendu permettant de construire une page avec des templates.*
- **Javascript** : *Langage de programmation côté client.*
- **HTML** : *Langage de balisage permettant de construire une page web.*
- **CSS** : *Langage permettant d'ajouter du style à la page.*

B. AJAX

1. Points client

- **Nom** : *getDataPoints*
- **Paramètres** :
  - ➢ **JSON** : *String*
- **Méthode** : *POST*
- **Type** : *JSON*
- **Code** :

```
function getDataPoints(json) {
	let data;
	$.ajax({
		async: false,
		type: "POST",
		url: "/app/widgets/{{ agence
}}/logistique/transport/gestion-des-tournees/map-gestion-tournees/data",
		timeout: 60000,
		data: {
			data: json
		},
		dataType: "json",
		success: function (result) {
			data = result;
		},
		error: function (req, err) {
			console.log("Error : " + err);
		}
	});
	return data;
}
```

2. Commandes par client

- **Nom** : *getDataClientCommandes*
- **Paramètres** :
  - ➢ **Date** : *String*
  - ➢ **Client** : *String*
- **Méthode** : *GET*
- **Type** : *JSON*
- **Code** :

```
function getDataClientCommandes(date, client) {
	let data;
	$.ajax({
		async: false,
		type: "GET",
		url: "/app/widgets/{{ agence
}}/logistique/transport/gestion-des-tournees/map-gestion-tournees/client
-commandes?date=" + date + "&client=" + client,
		dataType: "json",
		success: function (result) {
			data = result;
		},
		error: function (req, err) {
			console.log("Error : " + err);
		}
	});
	return data;
}
```

3. Tournée

- **Nom** : *getDataTournee*
- **Paramètres** :
  - ➢ **Tournee** : *String*
  - ➢ **Date** : *String*
  - ➢ **Clients** : String
- **Méthode** : *GET*
- **Type** : *JSON*
- **Code** :

```
function getDataTournee(tournee, date, clients) {
	clients = clients.join(',');
	let data;
	$.ajax({
		async: false,
		type: "GET",
```

```
            url: "/app/widgets/{{ agence
}}/logistique/transport/gestion-des-tournees/map-gestion-tournees/tourne
e?tournee=" + tournee + "&date=" + date + "&clients=" + clients,
            timeout: 60000,
            dataType: "json",
            success: function (result) {
                  data = result;
            },
            error: function (req, err) {
                  console.log("Error : " + err);
            }
      });
    return data;
}
```

4. Insérer une commande dans une tournée

- **Nom** : *insertCmdToTournee*
- **Paramètres** :
  - **Commande** : *String*
  - **Date** : *String*
  - **Plateforme** : String
  - **Tournee** : String
  - **positionTournee** : String
- **Méthode** : *GET*
- **Type** : *JSON*
- **Code** :

```
function insertCmdToTournee(commande, date, plateforme, tournee,
posTournee) {
      let agence = "{{ agence }}";
      let bp = 0;
      $("#map-popup-container-affectation").remove();
      $.ajax({
            async: false,
            type: "GET",
            url: ("/app/widgets/{{ agence
}}/logistique/transport/gestion-des-tournees/map-gestion-tournees/insert
-tournee?agence=" + agence.trim() + "&commande=" + commande.trim() +
"&date=" + date.trim() + "&plateforme=" + plateforme.trim() +
"&tournee=" + tournee.trim() + "&posTournee=" + posTournee + "&bp=" +
bp),
            timeout: 60000,
            dataType: "json",
            success: function (result) {
```

```
                if (result["code"] === 200) {
                        Swal.fire({
                                icon: "success",
                                title: "Affectation réussie avec succès",
                                toast: true,
                                position: "top-end",
                                showConfirmButton: false,
                                timer: 3000,
                                timerProgressBar: true
                        });
                } else {
                        Swal.fire({
                                icon: "error",
                                title: "Erreur suite à l'affection",
                                toast: true,
                                position: "top-end",
                                showConfirmButton: false,
                                timer: 3000,
                                timerProgressBar: true
                        });
                }
        },
        error: function (req, err) {
                console.log("Error : " + err);
                Swal.fire({
                        icon: "error",
                        title: "Erreur suite à l'affection",
                        toast: true,
                        position: "top-end",
                        showConfirmButton: false,
                        timer: 3000,
                        timerProgressBar: true
                });
        }
    });
    filter();
}
```

VI. Annexes

A. Points client

1. Contrôleur

```
public function getDataPoints(string $agence): void
{
    $connexion = $this->authentification($agence,
AuthMethodsEnum::BEARER);
    if (!$connexion) {
        return;
    }

    $data = json_decode(file_get_contents("php://input"), true);

    $this->createManager();
    $result = $this->_objManager->getDataPoints($agence, $data);

    if ($result === false) {
        PapHttpResponse::code500_erreurInterneServeur();
    } elseif (!empty($result)) {
        PapHttpResponse::code200_ok($result);
    } else {
        PapHttpResponse::code204_aucuneDonneesTrouvees();
    }
}
```

2. Manager

```
public function getDataPoints(string $agence, array $data): bool|array
    {
        $params = [
            "agence" => $agence,
            "dateLivraison" => $data["dateLivraison"]
        ];

        $where = " and ceage = ? and cedtld = ?";
        if ($data["tempSec"] || $data["tempFrais"] || $data["tempGel"])
{
            $where .= " and artem in (";
            $temp = [];
            if ($data["tempSec"]) {
                $temp[] = "'020'";
            }
            if ($data["tempFrais"]) {
```

```
                $temp[] = "'005'";
            }
            if ($data["tempGel"]) {
                $temp[] = "'-18'";
            }
            for ($i = 0; $i < sizeof($temp); $i++) {
                $where .= $temp[$i] . ($i < (sizeof($temp) - 1) ? ", " :
")");
            }
        }

        if (!empty($data["plateforme"])) {
            $where .= " and plateforme(ceage, cetol) = ?";
            $params["plateforme"] = $data["plateforme"];
        }
        if (!empty($data["tournee"])) {
            $where .= " and case bdnobp when 0 then cetol else bdtol end
= ?";
            $params["tournee"] = $data["tournee"];
        }
        if ($data["cmdBloquee"]) {
            $where .= " and case ceblq when 'O' then 1 else 0 end = ?";
            $params["cmdBloquee"] = ($data["cmdBloquee"] === "1" ??
"0");
        }
        if ($data["cmdAffectee"]) {
            $where .= " and bdtodt " . ($data["cmdAffectee"] === "1" ?
">" : "=") . " 0";
        }
        if(!empty($data["codeClient"])){
            $where .= " and ceccl = ?";
            $params["codeClient"] = $data["codeClient"];
        }
        if (!empty($data["numCommande"])) {
            $where .= " and cenum = ?";
            $params["cenum"] = $data["numCommande"];
        }
        $strRequest = "
with tournees as (
    select distinct
        ceccl,
        cedtld,
        (case total when 0 then cetol else bdtol end) as result
    from (
        select
            sum(bdnobp) as Total,
```

```
            cetol,
            bdtol,
            ceccl,
            cedtld
        from gcceen
        join gccede on cenum = cdnum and cefil = cdfil
        join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
        group by
            cetol, bdtol, ceccl, cedtld
    )
)

select
    ceage as Agence,
    ceccl as Client,
    acnom as Nom_Livraison,
    asbat as Batiment,
    asad1 as Adresse_1,
      asldi as Lieu_Dit,
    asad2 as Adresse_2,
      ascp concat ascp2 as Code_Postal,
    asvil as Ville,
    assta as Adresse_Stationnement,
    aslat as Latitude,
    aslon as Longitude,

    sum(case artem when '020' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Sec,
    sum(case artem when '005' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Frais,
    sum(case artem when '-18' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Gel,
    sum(cdqtuv * arlong * arlarg * arhaut / 1000000) as Volume,

    sum(case artem when '020' then cdpdbr else 0 end) as Poids_Sec,
    sum(case artem when '005' then cdpdbr else 0 end) as Poids_Frais,
    sum(case artem when '-18' then cdpdbr else 0 end) as Poids_Gel,
    sum(cdpdbr) as Poids,

    sum(case artem when '020' then 1 else 0 end) as NB_Lignes_Sec,
    sum(case artem when '005' then 1 else 0 end) as NB_Lignes_Frais,
    sum(case artem when '-18' then 1 else 0 end) as NB_Lignes_Gel,
    count(*) as NB_Lignes,

    (
    select
```

```
        listagg(trim(result), ',')
    from tournees t
    where t.ceccl = g.ceccl and t.cedtld = g.cedtld
    ) as tournees

from gcceen g
join gccede on cenum = cdnum and cefil = cdfil
join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
join gcarte on arartt = cdart
join ctadcl_t on aclie = ceccl and actad = 'LIV' and acdef = 'O' and
acannu = 'N'
join ctadres_t on acadr = asadr
where ceannu = 'N'
    and ceedt = 'N'
    and ceval = 'O'
    and cdann = 'N'
    " . $where . "

group by
    ceage,
    ceccl,
    acnom,
    asbat,
    asad1,
    asldi,
    asad2,
    ascp concat ascp2,
    asvil,
    assta,
    aslat,
    aslon,
    cedtld;
        ";

        try {
            return $this->_connexion->requeteDb($strRequest, $params,
true);
        } catch (Error $e) {
            return false;
        }
    }
```

B. Commandes par client

1. Contrôleur

```
public function getDataClientCommandes(string $agence): void
{
    $connexion = $this->authentification($agence,
AuthMethodsEnum::BEARER);
    if (!$connexion) {
        return;
    }

    $date = PapHttpRequest::getData("date");
    $client = PapHttpRequest::getData("client");

    $params = [
        "agence" => $agence,
        "date" => $date,
        "client" => $client
    ];

    $this->createManager();
    $result = $this->_objManager->getDataClientCommandes($params);

    if ($result === false) {
        PapHttpResponse::code500_erreurInterneServeur();
    } elseif (!empty($result)) {
        PapHttpResponse::code200_ok($result);
    } else {
        PapHttpResponse::code204_aucuneDonneesTrouvees();
    }
}
```

2. Manager

```
public function getDataClientCommandes(array $params): bool|array
    {
        $request = "
select
    cdccl as Client,
    cenum as Commande,
    (
    select distinct
        bdnobp
    from gcceen g2
```

```
        join gccede on cenum = cdnum and cefil = cdfil and cdann = 'N'
        join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
        join gcarte g3 on arartt = cdart
    where g2.ceannu = 'N'
        and g2.ceedt = 'N'
        and g2.ceval = 'O'
        and g2.ceage = 'MTB'
        and g2.cenum = g1.cenum
        and g3.artem = '020'
    ) as BP_Sec,
    (
    select distinct
        bdnobp
    from gcceen g2
        join gccede on cenum = cdnum and cefil = cdfil and cdann = 'N'
        join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
        join gcarte g3 on arartt = cdart
    where g2.ceannu = 'N'
        and g2.ceedt = 'N'
        and g2.ceval = 'O'
        and g2.ceage = 'MTB'
        and g2.cenum = g1.cenum
        and g3.artem = '005'
    ) as BP_Frais,
    (
    select distinct
        bdnobp
    from gcceen g2
        join gccede on cenum = cdnum and cefil = cdfil and cdann = 'N'
        join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
        join gcarte g3 on arartt = cdart
    where g2.ceannu = 'N'
        and g2.ceedt = 'N'
        and g2.ceval = 'O'
        and g2.ceage = 'MTB'
        and g2.cenum = g1.cenum
        and g3.artem = '-18'
    ) as BP_Gel,

    plateforme(ceage, cetol) as Plateforme,
    (
    select distinct
        case sum(bdnobp) when 0 then cetol else bdtol end
    from gcceen g2
    join gccede on cenum = cdnum and cefil = cdfil
    join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
```

```
    where g2.cenum = g1.cenum
    group by cetol, bdtol
    ) as Tournee,

    case ceblq when 'O' then 1 else 0 end as Bloquee,
    case sum(bdtodt) when 0 then 0 else 1 end as Affectee,

    equiv_pal(ceage, ceccl, cedtld, temp_cde(ceage, cenum, 'N', 'N'),
cenum, cemjad, 'N') as Equivalent_Palette,

    sum(case artem when '020' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Sec,
    sum(case artem when '005' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Frais,
    sum(case artem when '-18' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Gel,
    sum(cdqtuv * arlong * arlarg * arhaut / 1000000) as Volume,

    sum(case artem when '020' then cdpdbr else 0 end) as Poids_Sec,
    sum(case artem when '005' then cdpdbr else 0 end) as Poids_Frais,
    sum(case artem when '-18' then cdpdbr else 0 end) as Poids_Gel,
    sum(cdpdbr) as Poids,

    sum(case artem when '020' then 1 else 0 end) as NB_Lignes_Sec,
    sum(case artem when '005' then 1 else 0 end) as NB_Lignes_Frais,
    sum(case artem when '-18' then 1 else 0 end) as NB_Lignes_Gel,
    count(*) as NB_Lignes
from gcceen g1
join gccede on cenum = cdnum and cefil = cdfil
join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
join gcarte on arartt = cdart
where ceannu = 'N'
    and ceedt = 'N'
    and ceval = 'O'
    and cdann = 'N'
    and ceage = ?
    and cedtld = ?
    and cdccl = ?
group by
    cdccl,
    cenum,
    ceage,
    ceccl,
    cedtld,
    cemjad,
    cetol,
```

```
    bdtol,
    ceblq
order by cenum
        ";
        try {
            return $this->_connexion->requeteDb($request, $params,
true);
        } catch (Error $e) {
            return false;
        }
    }
```

C. Tournée

1. Contrôleur

```
public function getDataTournee(string $agence) {

   $connexion = $this->authentification($agence, AuthMethodsEnum::BEARER);

   if (!$connexion) {
       return;
   }

   $date = urldecode(PapHttpRequest::getData("date", ""));
   $tournee = urldecode(PapHttpRequest::getData("tournee" , ""));
   $clients = urldecode(PapHttpRequest::getData("clients", ""));

   $params = [
       "date" => $date,
       "tournee" => $tournee
   ];

   $clients = explode(",", $clients);

   foreach ($clients as $c) {
       $params[$c] = $c;
   }

   $this->createManager();
   $result = $this->_objManager->getDataTournee($params);

   if ($result === false) {
       PapHttpResponse::code500_erreurInterneServeur();
   } elseif (!empty($result)) {
       PapHttpResponse::code200_ok($result);
   } else {
       PapHttpResponse::code204_aucuneDonneesTrouvees();
   }

}
```

2. Manager

```
public function getDataTournee(array $params): bool|array
   {

       $clients = "(";
       for ($i = 0; $i < (sizeof($params) - 2); $i++) {
           $clients .= "?" . ($i == (sizeof($params) - 3) ? "" : ",");
       }
       $clients .= ")";
```

```
       $strRequest = "
select
   toage as Agence,
   toplt as Plateforme,
   totol as Tournee,

   tochf as Chauffeur,
   case when tochf <> ' ' then chnom else 'Inconnu' end as Chauffeur_Nom,

   totrs as Transporteur,
   case when tochf = ' ' and totrs <> ' ' then chnom else 'Inconnu' end as
Transporteur_Nom,

   tovehi as Vehicule,
   case when vhvehi is null then 0 else vhpal end as Vehicule_NB_Palette,
   case when vhvehi is null then 0 else vhpv end as Vehicule_PV,
   case when vhvehi is null then 0 else vhcu end as Vehicule_CU,
   case when vhvehi is null then 0 else vhvol end as Vehicule_Volume,
   case when lib_tycam is null then '' else lib_tycam end as Vehicule_Type,
   case when vhvehi is null then '-' else vhtemp end as Vehicule_Temperature,
   case when vhvehi is null then 'Pas de véhicule' else (trim(vhlib) concat case
when lib_tycam is null then '' else (' - ' concat lib_tycam) end) end as
Vehicule_libelle,
   case when vhvehi is null then 'Inconnu' else trim(inlib) end as
Vehicule_Temperature_Libelle,

   sum(Equivalent_Palette) as Equivalent_Palette,

   sum(Volume_Sec) as Volume_Sec,
   sum(Volume_Frais) as Volume_Frais,
   sum(Volume_Gel) as Volume_Gel,
   sum(Volume) as Volume,

   sum(Poids_Sec) as Poids_Sec,
   sum(Poids_Frais) as Poids_Frais,
   sum(Poids_Gel) as Poids_Gel,
   sum(Poids) as Poids,

   sum(NB_Lignes_Sec) as NB_Lignes_Sec,
   sum(NB_Lignes_Frais) as NB_Lignes_Frais,
   sum(NB_Lignes_Gel) as NB_Lignes_Gel,
   sum(NB_Lignes) as NB_Lignes,

   sum(case NB_Lignes_Sec when 0 then 0 else 1 end) as NB_Temp_Sec,
   sum(case NB_Lignes_Frais when 0 then 0 else 1 end) as NB_Temp_Frais,
   sum(case NB_Lignes_Gel when 0 then 0 else 1 end) as NB_Temp_Gel,
   count(*) as NB_Temp,

   listagg(case NB_Lignes_Sec when 0 then '' else (trim(ceccl) concat '-' concat
trim(cenum)) end, ',') as Client_Temp_Sec,
```

```
   listagg(case NB_Lignes_Frais when 0 then '' else (trim(ceccl) concat '-'
concat trim(cenum)) end, ',') as Client_Temp_Frais,
   listagg(case NB_Lignes_Gel when 0 then '' else (trim(ceccl) concat '-' concat
trim(cenum)) end, ',') as Client_Temp_Gel,
   listagg((trim(ceccl) concat '-' concat trim(cenum)), ',') as Client_Temp

from (
   select
       ceage,
       ceccl,
       cenum,
       cedtld,
       cetol,
       bdtol,
       sum(bdnobp) as Total,

       equiv_pal(ceage, ceccl, cedtld, temp_cde(ceage, cenum, 'N', 'N'), cenum,
cemjad, 'N') as Equivalent_Palette,

       sum(case artem when '020' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Sec,
       sum(case artem when '005' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Frais,
       sum(case artem when '-18' then (cdqtuv * arlong * arlarg * arhaut /
1000000) else 0 end) as Volume_Gel,
       sum(cdqtuv * arlong * arlarg * arhaut / 1000000) as Volume,

       sum(case artem when '020' then cdpdbr else 0 end) as Poids_Sec,
       sum(case artem when '005' then cdpdbr else 0 end) as Poids_Frais,
       sum(case artem when '-18' then cdpdbr else 0 end) as Poids_Gel,
       sum(cdpdbr) as Poids,

       sum(case artem when '020' then 1 else 0 end) as NB_Lignes_Sec,
       sum(case artem when '005' then 1 else 0 end) as NB_Lignes_Frais,
       sum(case artem when '-18' then 1 else 0 end) as NB_Lignes_Gel,
       count(*) as NB_Lignes

   from gcceen
   join gccede on cenum = cdnum and cefil = cdfil
   join gcbpde on cdnum = bdnum and cdfil = bdfil and cdlig = bdlig
   join gcarte on arartt = cdart

   where ceannu = 'N'
       and ceedt = 'N'
       and ceval = 'O'
       and cdann = 'N'

   group by
       ceage,
       ceccl,
       cenum,
```

```
        cedtld,
        cetol,
        bdtol,
        cemjad
)
join gctol on totol = case Total when 0 then cetol else bdtol end and totodt =
cedtld
left join gcvehi on tovehi = vhvehi
left join gcchf on case when tochf <> ' ' then tochf else totrs end = chchf
left join dic_tycam on code_tycam = vhtyp and supr_tycam = 'N'
left join dcint on incod = 'TECA' and inarg = vhtemp and insup = 'N' and intitr
= 'N'

where totodt = ?
   and totol = ?
   and ceccl in " . $clients . "

group by
   toage,
   toplt,
   totol,
   tochf,
   chnom,
   totrs,
   tovehi,
   vhvehi,
   vhpal,
   vhpv,
   vhcu,
   vhvol,
   lib_tycam,
   vhtemp,
   vhlib,
   inlib
        ";

        try {
            return $this->_connexion->requeteDb($strRequest, $params, true);
        } catch (Error $e) {
            return false;
        }

   }
```

D. Plateformes et tournées

1. Contrôleur

```
public function getDataPlateformesAndTournees(string $agence) {

   $connexion = $this->authentification($agence,
AuthMethodsEnum::BEARER);

   if (!$connexion) {
       return;
   }

   $params = [
       "agence" => $agence
   ];

   $this->createManager();
   $result = $this->_objManager->getDataPlateformesAndTournees($params);

   if ($result === false) {
       PapHttpResponse::code500_erreurInterneServeur();
   } elseif (!empty($result)) {
       PapHttpResponse::code200_ok($result);
   } else {
       PapHttpResponse::code204_aucuneDonneesTrouvees();
   }

}
```

2. Manager

```
public function getDataPlateformesAndTournees(array $params): bool|array
{

   $strRequest = "select distinct
                       trim(code_plth) as code_plateforme,
                       trim(lib_plth) as nom_plateforme,
                       (trim(code_plth) concat case when trim(lib_plth) = ''
then '' else (' - ' concat trim(lib_plth)) end) as libelle_plateforme,
                       trim(tltol) as code_tournee,
                       trim(tltxt) as nom_tournee,
                       (trim(tltol) concat case when trim(tltxt) = '' then ''
else (' - ' concat trim(tltxt)) end) as libelle_tournee
                   from gcto
                   right join dic_plth on code_plth = tlplt and supr_plth = 'N'
                   where tlage = ?
                   and tlann = 'N'
```

```
                    order by trim(code_plth), trim(tltol);";

    try {
        return $this->_connexion->requeteDb($strRequest, $params, true);
    } catch (Error $e) {
        return false;
    }

}
```

E. Insérer une commande dans une tournée

1. Contrôleur

```
public function insertCmdToTournee(string $agence) {

   $connexion = $this->authentification($agence,
AuthMethodsEnum::BEARER);

   if (!$connexion) {
       return;
   }

   $agence = urldecode(PapHttpRequest::getData("agence", ""));
   $commande = intval(urldecode(PapHttpRequest::getData("commande",
0")));
   $date = urldecode(PapHttpRequest::getData("date"));
   $plateforme = urldecode(PapHttpRequest::getData("plateforme", ""));
   $tournee = urldecode(PapHttpRequest::getData("tournee", ""));
   $posTournee = intval(urldecode(PapHttpRequest::getData("posTournee",
0")));
   $bp = intval(urldecode(PapHttpRequest::getData("bp", 0")));

   $paramsIn = [
       "I_Optimiseur" => "EGESCOM",
       "I_Agence" => $agence,
       "I_NoCde" => $commande,
       "I_NoBp" => $bp,
       "I_DateTournee" => $date,
       "I_Tournee" => $tournee,
       "I_NoTournee" => 0,
       "I_PosTournee" => $posTournee,
       "I_Plateforme" => $plateforme
   ];
   $paramsOut = [
       "O_CodRet" => "",
       "O_CodAno" => "",
       "O_MsgAno" => ""
   ];

   $this->createManager();
   $result = $this->_objManager->insertCmdToTournee($paramsIn,
$paramsOut);

   if ($result === false) {
       PapHttpResponse::code500_erreurInterneServeur();
   } elseif (!empty($result)) {
```

```
        PapHttpResponse::code200_ok($result);
    } else {
        PapHttpResponse::code204_aucuneDonneesTrouvees();
    }

}
```

2. Manager

```
public function insertCmdToTournee($paramsIn, $paramsOut): bool|array
{

    $strRequest = "CALL PR_CTAFCE (cast(? as char(10)), cast(? as
                    char(3)), cast(? as numeric(7, 0)), cast(? as numeric(7, 0)),
                    cast(? as date), cast(? as char(3)), cast(? as
                    numeric(2, 0)), cast(? as decimal(2, 0)), cast(? as char(3)),
                    cast(? as char(2)), cast(? as char(2)), cast(? as
char(66)))";

    try {
        return $this->_connexion->requeteCall($strRequest, $paramsIn,
$paramsOut);
    } catch (Error $e) {
        return false;
    }

}
```