



# TPI RUNEO DRIVE

Clément Sartoni - 2023

## Résumé

TPI consistant en la continuation du projet Runeo Drive, sujet de plusieurs projets d'entraînement depuis janvier. Le projet Runeo dans son ensemble permet aux chauffeurs de paléo de gérer les déplacements des artistes et de leur entourage lors du festival

# 1. Analyse préliminaire

## 1.1. Introduction

Ce projet a été initié par M. Carrel il y a 4 ans pour répondre aux besoins d'organisation des chauffeurs du Paléo. Il a été repris par trois élèves de la classe min-mid 4 de l'etml dans le cadre de la préparation au TPI, puis du présent TPI. Une méthodologie agile sera utilisée et appliquée grâce à l'outil IceScrum, qui servira aussi de journal de travail.

L'application a déjà fait ses preuves et est déjà utilisée au paléo depuis plusieurs années. Elle est séparée en deux parties : l'application de bureau utilisée par les coordinateurs pour planifier les trajets, et l'application mobile utilisée par les chauffeurs pour recevoir les informations et prendre en charge les trajets. Ce projet se focalisera sur l'application mobile et les fonctionnalités à ajouter dans celle-ci.

Ce rapport détaillera l'analyse, la réalisation ainsi que le bilan de la réalisation de ces fonctionnalités.

## 1.2. Objectifs

### 1.2.1. Page de profil

Accédée à partir du sommet de la liste des chauffeurs, cette page montre les informations de profil de l'utilisateur connecté:

- Photo
- Prénom, nom et nom d'affichage (pseudo)
- Email
- Groupe
- Rôle
- Statut

### 1.2.2. Correction de statut

Dans le cas où je constate dans ma page de profil que mon statut est incorrect, l'application me permet de le mettre à jour au moyen d'une liste déroulante dont les éléments sont fournis par le backend.

### 1.2.3. Mes horaires et mes runs

Accédée à partir du sommet de la liste des chauffeurs, cette page montre les horaires de mon groupe dans une vue de type calendrier.

Cette vue est continue et verticale, c'est-à-dire qu'on peut scroller verticalement de manière continue pour voir les horaires précédents et suivants.

A l'intérieur de chaque horaire figurent les runs qui me concernent. La maquette ci-contre n'est qu'une suggestion de mise en page possible.

Figure 1 : Exemple de disposition d'horaire

La couleur des runs doit refléter leur état (futur, en cours ou terminé).

Pour chaque run, on a :

- Obligatoirement son numéro
- Obligatoirement une icône « multiple » s'il y a plus d'un véhicule dans le run (p.ex : )
- Le nom de l'artiste si on a la place
- Le nom du run si on a encore de la place
- Le nom du ou des autres chauffeurs s'il y en a et qu'il y a de la place pour les noms

L'application doit montrer les informations obligatoires et autant d'information optionnelle que possible en fonction de l'espace disponible. Elle peut choisir de supprimer ou d'écourter une information optionnelle trop longue.



#### 1.2.4. Gestion d'erreur

L'application est fortement dépendante de l'API offerte par le backend, laquelle peut être parfois indisponible ou ne pas fonctionner correctement.

L'application doit prendre en compte les divers cas d'erreurs possibles (pas de réseau, pas de réponse, code d'erreur http, ...) et les traiter proprement d'un point de vue UX : messages en français, non techniques, timeouts, ...)

### 1.3. Gestion de projet

Pour ce projet, la méthodologie Agile est utilisée. La majorité des éléments principaux ont été appliqués :

- Organisation du travail en sprints et user stories
- Daily meetings
- Sprint reviews pour valider le travail
- Journal de travail sous la forme de progression dans les tâches planifiées pour chaque user story.

Pour ce faire, l'outil IceScrum a été utilisé. C'est une application en ligne permettant de gérer un projet Agile de A à Z en créant des sprints, des user stories, et des tâches qui vont avec pour enregistrer le temps passé sur chacune d'entre elles. Le journal de travail peut ensuite être généré grâce à l'API de l'outil et un fichier html réalisé par le chef de projet M. Carrel.

## 1.4. Planification initiale

Le projet a débuté le lundi 1<sup>er</sup> mai à 8h et se terminera le mercredi 31 mai 2023 à 12h15.

Dans une semaine de travail normale, le temps disponible pour travailler est organisé comme indiqué dans ce tableau (toutes les demi-journées ayant une pause obligatoire de 15 minutes) :

Jour	Lundi	Mardi	Mercredi	Jeudi	Vendredi	Total
Horaire	08:00 - 11:25 13:10 - 16:35	-	08:00 - 12:15 13:10 - 16:35	13:10 - 16:35	08:00 - 12:15 13:10 - 16:35	
Heures	6h20	0h	7h10	3h10	7h10	23h50

Il y a trois jours fériés durant le projet : l'Ascension le jeudi 18 mai, son pont le vendredi 19 mai ainsi que le lundi de pentecôte le 29 mai 2023.

En prenant donc en compte les vacances et les jours de congé, il y a à disposition pour ce projet 89 heures.

### 1.4.1. Découpe en sprints

Vu le peu de temps à disposition pour ce projet, il a été choisi d'effectuer des sprints d'une semaine afin d'avoir un suivi plus pertinent et efficace. Cela a le désavantage de passer plus de temps sur des aspects organisationnels mais cela est nécessaire pour ne pas manquer de retours sur le travail qui est en cours, et si le travail de gestion des sprints est fait efficacement le coût se limite à quelques heures par semaine. Les sprints reviews se feront le jeudi à 15h00, et les sprints commenceront donc le vendredi matin de chaque semaine. Le dernier vendredi sera consacré et aux dernières retouches qui seront potentiellement discutées lors de la dernière sprint review et la dernière matinée du mercredi 31 sera dédiée à l'impression du rapport et à son envoi.

#### 1.4.1.1. Sprint 1

Les objectifs de ce sprint sont les suivants :

- La partie analyse du rapport (points 1 et 2) est terminée.
- Les sprints ainsi que les user stories sont définis sur IceScrum. Ceci inclut pour chaque user story une description détaillée, les tests d'acceptance ainsi qu'une première ébauche des tâches à réaliser.
- L'organisation des pages à ajouter a été discutée et clarifiée avec le chef de projet.
- Commencer la recherche et les tests pour l'affichage de l'horaire.

La sprint review se fera le jeudi 4 mai 2023 à 15h00.

#### 1.4.1.2. Sprint 2

L'objectif de ce sprint sera principalement de réaliser la user story concernant l'affichage de l'horaire de l'utilisateur et de ses runs, avec la gestion d'erreur associée. Celle-ci a été choisie comme première tâche

car c'est potentiellement la plus complexe à réaliser, étant donné que le sprint 2 est le seul sprint que ne sera pas perturbé par l'Ascension.

La sprint review se fera le jeudi 11 mai 2023 à 15h00.

#### *1.4.1.3. Sprint 3*

L'objectif de ce sprint sera de réaliser la user story concernant la page de profil de l'utilisateur, de commencer celle de la correction du statut ainsi que de documenter la réalisation du travail effectué jusqu'à ce point (points de design spécifiques).

La sprint review se fera le mercredi 17 mai 2023 à 15h00.

#### *1.4.1.4. Sprint 4*

Pour ce dernier sprint, l'objectif sera de finaliser la fonctionnalité de correction du status, de la documenter et de conclure le rapport (réalisation, bilan, conclusion, glossaire, etc.)

La sprint review se fera le jeudi 25 mai 2023 à 15h00, afin d'avoir encore une journée de travail pour effectuer des retouches avant le TPI si nécessaire ainsi qu'une matinée pour effectuer la livraison finale.

## 2. Analyse / Conception

### 2.1. Analyse fonctionnelle

#### 2.1.1. Page de mon horaire et mes runs

En tant que chauffeur, je veux pouvoir visualiser mes horaires ainsi que les runs que j'ai à faire dans une vue de style calendrier, afin de pouvoir mieux gérer mon temps et d'avoir accès facilement à ces informations importantes.

Tests d'acceptance :

<b>Chargement</b>	Une fois que l'on est connecté, en accédant à la page "horaires", les données de mes horaires et de mes runs (passés ou futurs) se chargent sur la page (voir maquettes).
<b>Position actuelle</b>	Une fois la page horaires chargée, je vois une barre violette indiquant l'heure actuelle aux 3/4 de l'écran, et les autres éléments de l'horaire sont à la bonne position par rapport à l'heure.
<b>Scroll</b>	Sur la page horaires, je peux scroller vers le passé et le futur jusqu'au premier et dernier jour pour lequel mon groupe a des tranches horaires prévues, et tous les runs que j'ai effectués ou qui sont prévus sont affichés.
<b>Affichage d'un run au nom court</b>	Sur la page horaires, les runs simples sont affichés avec leur numéro ainsi que leur titre.
<b>Affichage d'un run au nom long</b>	Sur la page horaires avec un run dont le nom est trop long, le run est affiché avec son numéro, le début du titre et trois petits points.
<b>Affichage d'un run multiple</b>	Sur la page horaires, les runs à plusieurs chauffeurs sont affichés avec une icône dédiée ainsi que le nom des autres chauffeurs participant connus.
<b>Clic sur un run</b>	Sur la page horaires, en cliquant sur un run, on est redirigé vers sa page de détails.
<b>Retour depuis un run</b>	Sur la page d'un run ouvert depuis la page horaires, en cliquant sur "retour", on arrive de nouveau sur la page horaires.
<b>Couleurs des runs</b>	Sur la page horaires, la couleur d'un run dépend de son statut. (Voir 'status colors')
<b>Date du jour</b>	Sur la page horaires, la date est indiquée en haut à gauche. Quand le jour suivant arrive dans le quart supérieur de l'écran, la date passe au jour suivant.
<b>Erreur: pas de data</b>	Quand j'essaie d'accéder à la page horaires mais que je ne suis pas connecté à internet (pas de data ou mode avion), un message d'erreur apparaît pour me signaler que je ne suis pas connecté et me demander de vérifier ma connexion ou de réessayer plus tard.

<b>Erreur: pas de réponse du backend</b>	Quand j'essaie d'accéder à la page horaires mais que le serveur ne répond pas après quelques secondes, un message d'erreur apparaît pour me signaler que les données n'ont pas pu être chargées et me demander de réessayer plus tard.
<b>Erreur: 0x4xx ou 0x5xx</b>	Quand j'essaie d'accéder à la page horaires mais que le serveur retourne une erreur, un message d'erreur apparaît pour me signaler qu'il y a eu un problème et me demander de contacter un administrateur.
<b>No cache</b>	Quand je suis sur la page d'horaires depuis quelques secondes et que le statut d'un de mes runs a été changé depuis sur le backend, que je quitte la page d'heure et que j'y reviens, le statut est bien celui du backend (nouveau).

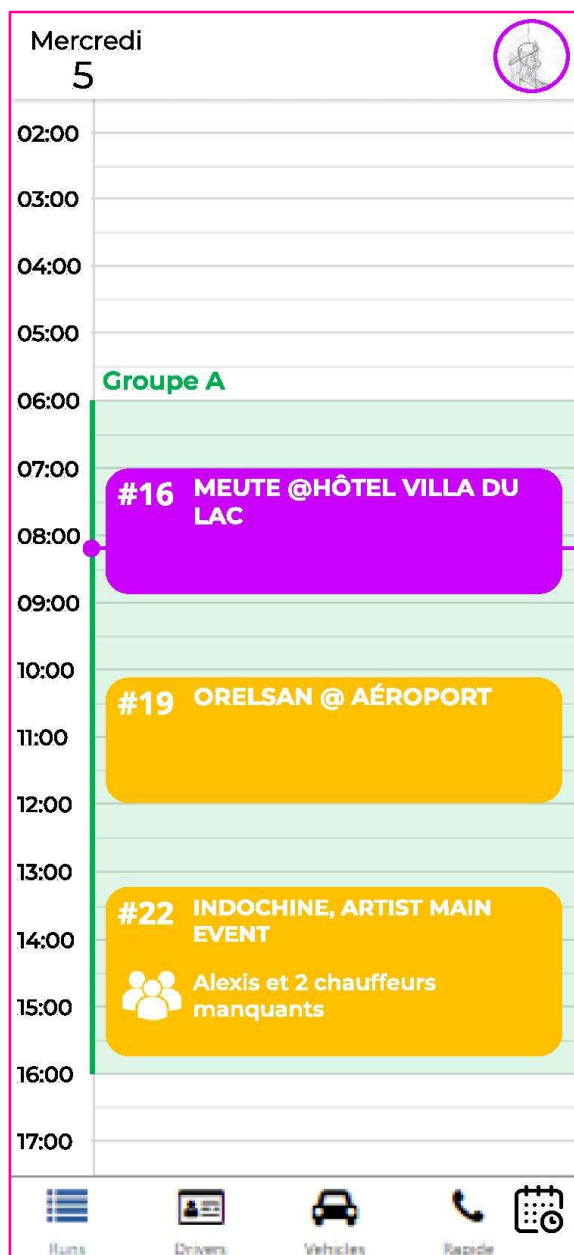


Figure 3: Maquette "En Run"

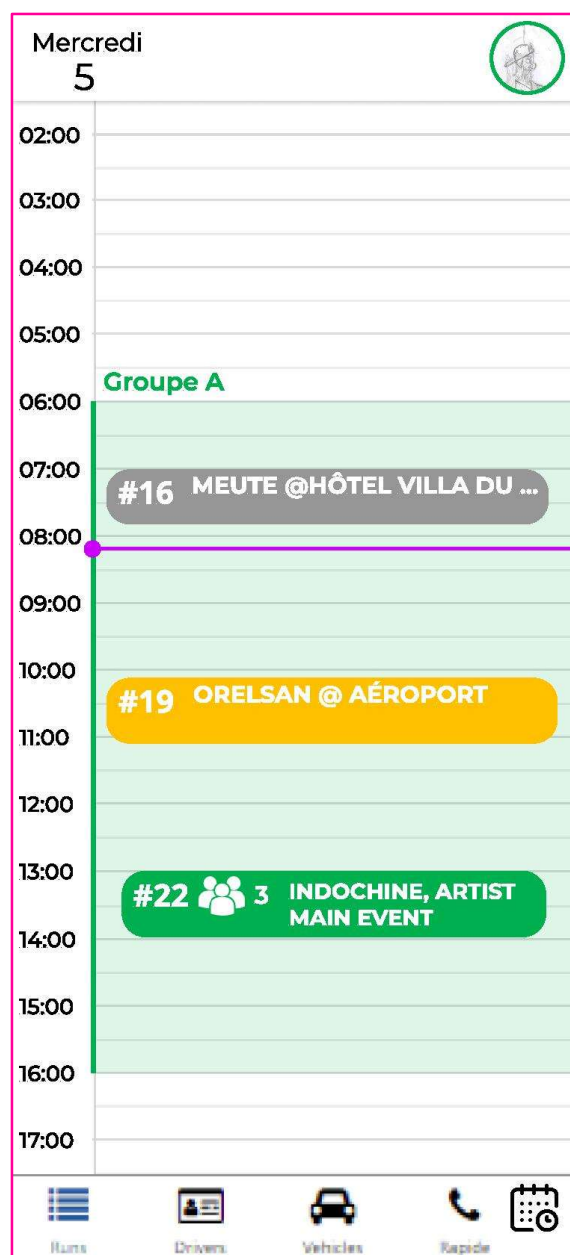


Figure 2 : Maquette "Disponible Small Runs"



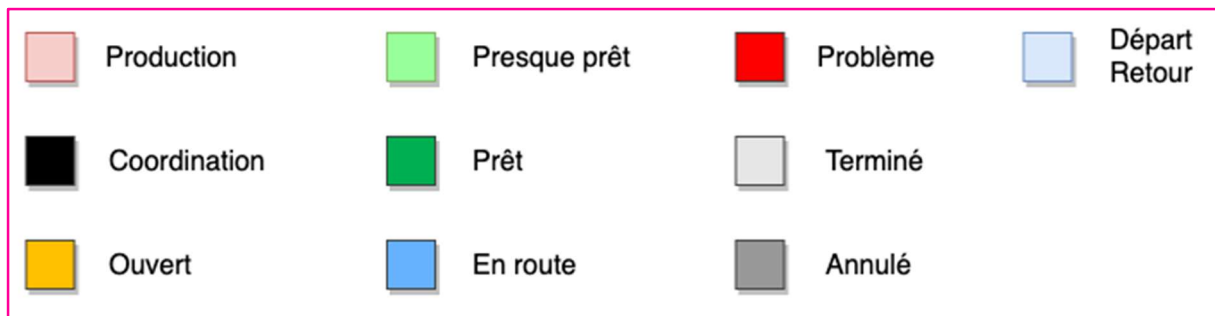


Figure 4 : "Status Colors", couleurs actuellement utilisées dans l'application

### 2.1.1.1. Chargement des données depuis l'API

Cache ou pas cache ? Mettre ce point ici ou pas ? que de questions et si peu de réponses, heureusement qu'on pourra y répondre demain.

### 2.1.2. Page de mon profil

En tant que chauffeur, je veux avoir accès aux informations de mon profil, afin de pouvoir les vérifier et modifier mon statut.

<b>Accès à la page</b>	Lorsque je suis sur la page de mes horaires, en cliquant sur ma photo de profil, je suis dirigé vers la page de mon profil. (Voir maquette)
<b>Paramètres</b>	Sur la page de mon profil, en cliquant sur la roue crantée en haut à droite, je suis redirigé vers la page de paramètres actuelle.
<b>Retour en arrière</b>	Sur la page de profil, en cliquant sur la flèche de retour en haut à gauche, je retourne à la page de mes horaires.
<b>Dropdown de mon statut</b>	Sur la page de mon profil, en cliquant sur mon statut, une dropdown s'ouvre et je peux choisir entre les différentes possibilités (engagé, disponible, en run, indisponible).
<b>Modification de mon statut</b>	Après avoir ouvert la dropdown des statuts, quand je clique sur l'un des statuts, le nom du statut actuel change et la couleur autour de l'avatar aussi.
<b>Gestion d'erreur permanente</b>	Après avoir cliqué pour modifier mon statut, si la requête de modification a échoué (http erreur 0x4xx ou 0x5xx), un message d'erreur m'avertit que la modification a échoué et que je dois contacter un administrateur.
<b>Gestion d'erreur transitoire</b>	Après avoir cliqué pour modifier mon statut, si je ne reçois pas de réponse après quelques secondes, un message d'erreur m'avertit que la modification a échoué, que je ne suis probablement pas connecté et me demande de réessayer plus tard.
<b>Couleur du statut</b>	Sur la page de mon profil, chaque statut est associé à une couleur, affichée sur le texte et autour de ma photo de profil (voir image couleurs status).
<b>No cache</b>	Quand je suis sur la page de profil depuis quelques secondes et que mon statut a été changé depuis sur le backend, quand je quitte la page de profil et que j'y reviens, le statut est bien celui du backend (nouveau).

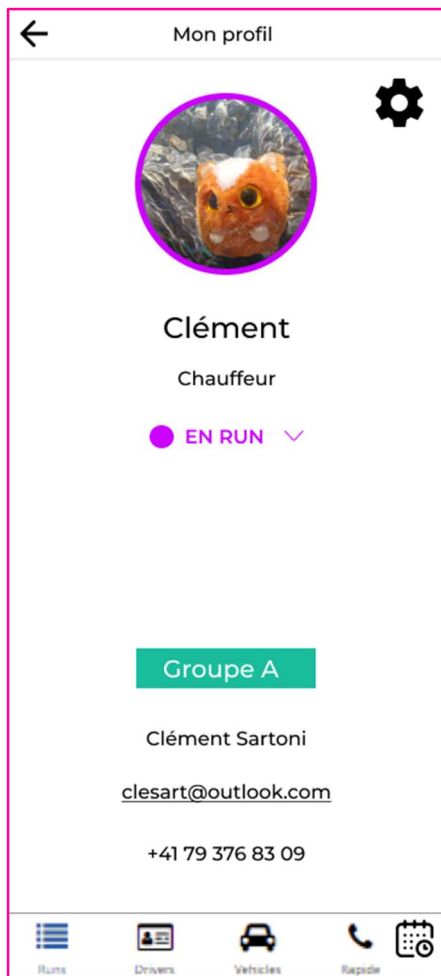


Figure 5 : Maquette page profil

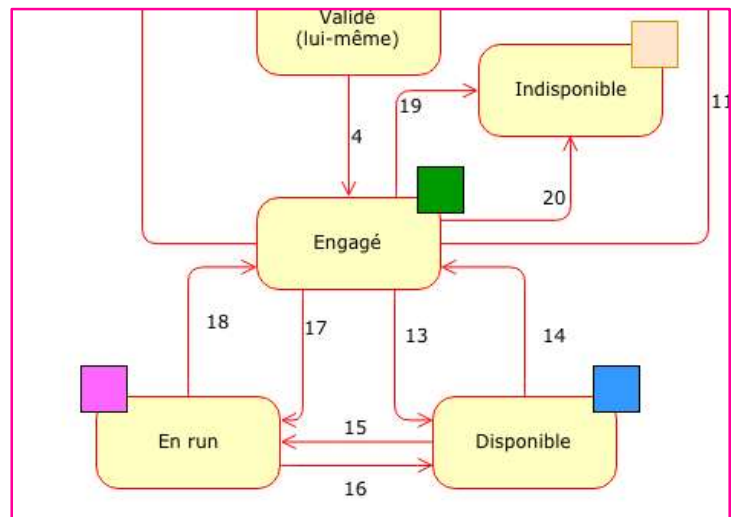


Figure 6 : Couleurs des statuts des chauffeurs actuellement utilisées

## 2.2. Concept

Le but premier de l'application est de gérer les runs, ou courses, qui sont concrètement les demandes ou besoins de transport des artistes ou des personnes les accompagnant. Ces runs sont planifiés par les coordinateurs soit avant le festival soit pendant, avec toutes les informations nécessaires comme les heures de rendez-vous et le type de véhicule par exemple. Ensuite, ils apparaissent sur la page principale de l'application mobile et les chauffeurs peuvent se désigner pour effectuer les runs tout en sélectionnant leur véhicule.

### 2.2.1. Infrastructure

Le site web, codé en utilisant le framework Laravel, est directement utilisé par les coordinateurs. Il permet de gérer toutes les données, qui sont stockées dans une base de données MySQL. Il met à disposition une API pour que l'application mobile, réalisée en React Native et installées sur les smartphones des chauffeurs, puisse récupérer les informations. Les chauffeurs se connectent à l'application mobile à l'aide d'un token lié à leur compte, et les coordinateurs se connectent sur le site web avec une combinaison classique d'email et de mot de passe.

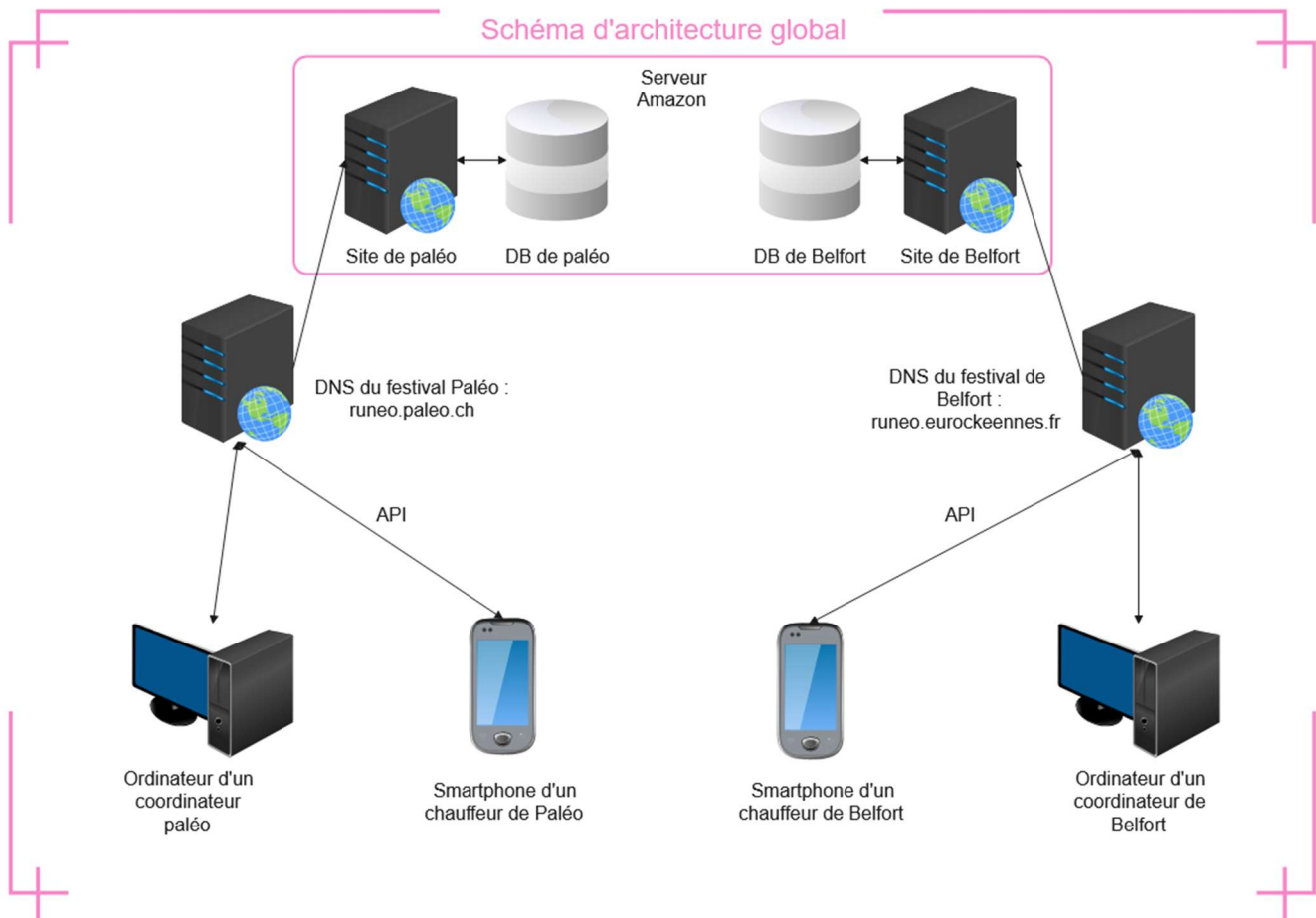


Figure 7 : Schéma expliquant le fonctionnement de l'architecture

### 2.2.2. Données

La base de données n'a pas été modifiée durant ce projet. Elle a été créée lors du début du projet il y a plusieurs années par d'autres personnes, et le MCD et le MLD qui seront présentés dans ce rapport aussi. Les explications se focaliseront ici sur quelques tables qui ont été utilisées durant ce projet. Les fichiers complets sont disponibles en annexe.

runs	
123 id	int(10) unsigned NOT NULL
ABC name	varchar(191)
🕒 published_at	datetime
🕒 planned_at	datetime
🕒 end_planned_at	datetime
🕒 started_at	datetime
🕒 ended_at	datetime
123 passengers	int(11)
ABC infos	longtext
ABC prodid	varchar(60)
🕒 deleted_at	timestamp
🕒 created_at	timestamp
🕒 updated_at	timestamp
123 pax_tbc	tinyint(4) NOT NULL
123 time_tbc	tinyint(4) NOT NULL
123 status_id	int(10) unsigned
123 coordinator_estimation_minutes	bigint(20)
ABC transportationMeans	text
ABC nameContact	text
ABC numContact	text
123 artist_id	int(10) unsigned

Pour commencer, voici le MLD de l'entité « runs ». Elle représente une course qu'un chauffeur doit effectuer pour un artiste.

Le champ « name » est compilé à partir du nom de l'artiste et du point de rendez-vous (p. ex. « Artiste @ Aéroport »).

Le run est relié à l'artiste concerné via la foreign key « artist\_id ».

Pour savoir qui va prendre en charge le run et quelle voiture sera utilisée, une autre table « run\_drivers » a été créée. Elle était nécessaire étant donné qu'un run peut nécessiter plusieurs paires voitures-chauffeurs.

run_drivers	
123 id	int(10) unsigned NOT NULL
123 user_id	int(10) unsigned
123 run_id	int(10) unsigned
123 car_id	int(10) unsigned
123 car_type_id	int(10) unsigned
ABC status	varchar(191) NOT NULL
🕒 deleted_at	timestamp
🕒 created_at	timestamp
🕒 updated_at	timestamp
🕒 acknowledged_at	datetime

Pour le festival de paléo, il est souvent précisé quel type de véhicule est requis (champ « car\_type\_id »), puis les chauffeurs peuvent s'inscrire via l'application mobile et choisir leur véhicule plus précisément.

Figure 8 : MCD de la base de données

Pour transmettre ces informations à l'application mobile, l'API du site web les formate et combine d'une manière qui soit plus pratique à gérer, en format JSON.

```
▼ [Run ▼ {  
  id                integer  
  status            string  
  title             string  
  begin_at          string  
  start_at          string  
  updated_at        string  
  acknowledged_at   string  
  pax_tbc           integer  
  time_tbc          integer  
  end_at            string  
  finished_at       string  
  nb_passenger      string  
  runinfo           string  
  name_contact      string  
  num_contact       string  
  waypoints         ▼ [Waypoint ▼ {  
    nickname        string  
    meeting_time    string  
  }]  
  runners           ▼ [Runner ▼ {  
    id              integer  
    user            User > {...}  
    vehicle_category CarType > {...}  
    vehicle         Car > {...}  
  }]  
  artist_id         integer  
}]
```

Voici la vision schématique de l'objet JSON récupéré de l'API depuis l'application mobile, générée par l'application Swagger.

On peut voir que seules les informations utiles aux chauffeurs sont transférées (même si presque toutes sont là), et que toutes les informations des autres tables comme les points de passages ou les informations des chauffeurs sont aussi passées dans le même objet.

On peut aussi voir le champ « artist\_id » qui est celui qui a été ajouté pour les besoins de la fonctionnalité de la page « autres runs de l'artiste ».

## 2.2.3. Cycle de vie d'un run

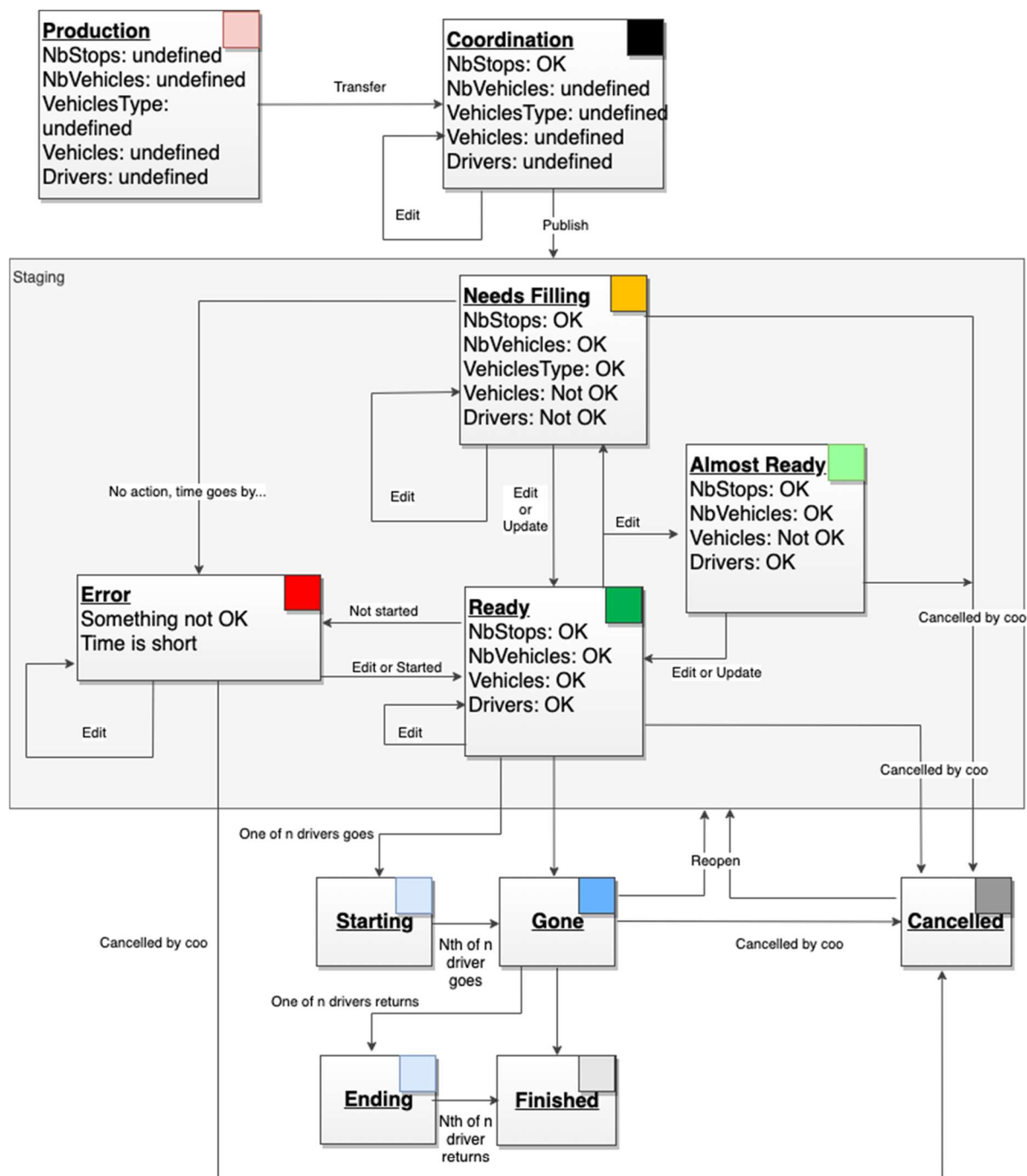
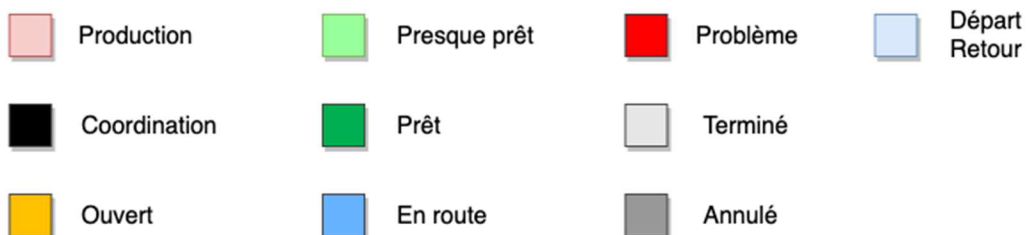
**Textes d'affichage**

Figure 9: Schéma détaillant les différents états par lesquels passe un run

Ce schéma, créé lui aussi par d'autres personnes lors de la création du projet, détaille tout le cycle de vie d'un run, de sa création à son exécution ou son annulation.

Tout d'abord, la production transfère à la coordination un run potentiellement incomplet, avec les premières informations comme la date et la description. Ensuite, la coordination s'assure de renseigner au moins le point de rendez-vous, l'heure ainsi que le type et le nombre de voitures nécessaires avant de rendre le run disponible aux chauffeurs en le passant en mode « staging ».

Ensuite, Les chauffeurs peuvent prendre le run et sélectionner leur véhicule. Si personne ne s'inscrit à temps ou s'il y a d'autres problèmes, le run est indiqué comme ayant des problèmes pour attirer l'attention des coordinateurs. Puis les chauffeurs démarrent le run sur l'application et le terminent lors de la fin de la course.

Durant tout le processus, il a été prévu que les utilisateurs ne risquent jamais d'être bloqués par le système. Il est à tout moment possible pour les coordinateurs d'assigner un chauffeur à un run et de le démarrer de force si le chauffeur est déjà parti sans effectuer les actions dans son application mobile par exemple.

## 2.2.4. Fonctionnement de l'application mobile

Les chauffeurs se font envoyer un lien de téléchargement via WhatsApp ou téléchargent l'application depuis le Google Play Store. Ils reçoivent aussi de la part des coordinateurs un token de connexion leur permettant de s'identifier sur la première page de l'application et d'avoir accès aux runs.



## 2.2.5. Arborescence des fichiers de l'application mobile



Voici l'arborescence de tous les fichiers du dossier « src », situé à la racine du projet. C'est ici que sont situés tous les fichiers de code.

Dans « common » se trouvent tous les composants dont l'appel peut être utile à plusieurs endroits de l'application. Il y a aussi un dossier par page principale (runs, users, vehicles, ...)



## 2.3. Stratégie de test

En raison d'un problème concernant le build de l'application, il ne sera pas possible pour ce projet d'appliquer la procédure de test qui serait idéale et devrait normalement être utilisée. Ce chapitre sera donc séparé en deux.

### 2.3.1. Stratégie de test théorique

Il faudrait normalement installer complètement l'application sur un smartphone Android et sur un iPhone à chaque session de test pour vérifier que les modifications fonctionnent sur un environnement semblable à celui qui sera utilisé en production.

Pour ce faire, il est théoriquement possible de build l'application puis de la distribuer sur un canal de test sur le Play Store (Android) ainsi que sur l'App Store (iOS). Cependant, il est pour cela nécessaire d'avoir auparavant configuré des comptes permettant de publier des applications sur les deux systèmes. Dans l'état actuel, le compte développeur sur le Play Store est presque configuré et prêt à publier l'application, mais il n'y a rien de prévu pour iOS.

### 2.3.2. Stratégie de test utilisée

À chaque sprint review, M. Carrel effectuera un pull de la branche « develop » de GitHub sur son poste. Ensuite, l'application sera lancée en utilisant l'application « Expo Go » sur son smartphone. Les tests mentionnés dans l'analyse des user stories qui auront été terminées pourront ensuite être effectués.

Changer ainsi d'environnement permet de vérifier que les modifications effectuées dans l'environnement de développement fonctionnent une fois déployées ailleurs.

Concernant les données de test, tant que personne n'utilise la base de données de production de paléo, donc tant que les données de l'année passée n'ont pas été effacées, il est possible de créer des runs de tests depuis le site web afin d'utiliser le même backend qu'en production.

Il est possible que dans le futur la production ne soit plus disponible pour des tests, ce qui impliquerait de devoir lancer en plus de l'application mobile l'application Runeo-Desk localement et d'utiliser une base de données locale dans laquelle des runs de test pourront être créés.

## 2.4. Risques techniques

Les risques techniques sont assez légers pour ce projet. Le principal est le manque de formation concernant le React et son fonctionnement dans une application en React Native, qui est le framework utilisé dans l'application mobile. Cependant, cela reste un langage connu et relativement maîtrisé à la suite d'un projet précédent sur la même application.

Un autre risque potentiel est celui de reprendre du code déjà existant et codé par d'autres personnes. Cela rend plus complexe de prédire combien de temps la réalisation des fonctionnalités va prendre, étant donné que selon la manière dont l'application a été réalisée, il sera possible de plus ou moins réutiliser des éléments ou alors il pourrait ne pas être possible en l'état d'ajouter la fonctionnalité ce qui demanderait de remodeler des fonctionnalités déjà existantes.

## 2.5. Choix techniques

Ces choix techniques ont été effectués lors de la création du projet et aucune modification n'a donc été appliquée à ces derniers.

Outil de gestion des versions	Github
Framework du site web Runeo Desk	Laravel 9.16.0
Framework JS de l'application mobile	React Native version 14.4
Kit de développement logiciel de l'application mobile	Expo SDK version 43

## 5.2. Sources – Bibliographie

Description	Lien(s)
Outil utilisé pour réaliser les maquettes	<a href="https://www.figma.com">www.figma.com</a>
Outil pour transformer un PDF en JPG	<a href="https://pdf2jpg.net">https://pdf2jpg.net</a>
Sources des icônes utilisées dans les maquettes	<a href="https://www.flaticon.com/free-icon/multiple-users-silhouette_33308">https://www.flaticon.com/free-icon/multiple-users-silhouette_33308</a> <a href="https://www.flaticon.com/free-icon/settings-cogwheel-button_61094">https://www.flaticon.com/free-icon/settings-cogwheel-button_61094</a>
Utilisé pour la correction d'un bug dans le journal de travail	<a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Intl/DateTimeFormat/DateTimeFormat#options">https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Intl/DateTimeFormat/DateTimeFormat#options</a>
Documentation de la librairie JS moment me servant à gérer certaines dates et heures	<a href="https://momentjs.com/docs/#/displaying/">https://momentjs.com/docs/#/displaying/</a>

## 5.3. Journal de travail

## 5.4. Schémas complets

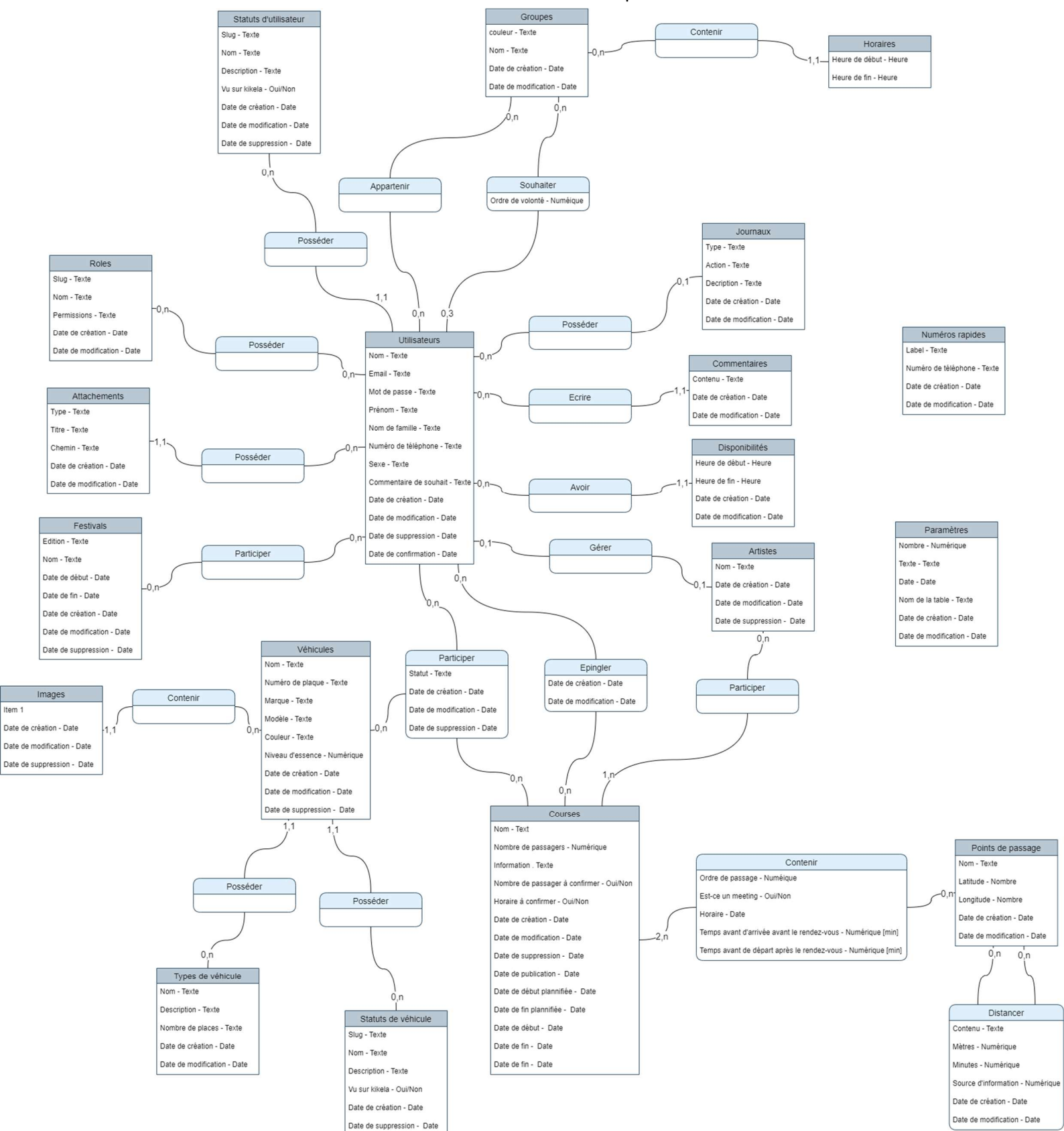










Figure 40 : MLD des données, créé durant lors de la création du projet il y a plusieurs années.

## 5.5. Cahier des charges

**1 INFORMATIONS GENERALES**

Candidat :	Nom : <b>SARTONI</b>	Prénom : <b>CLÉMENT</b>																				
	 : Clement.Sartoni@eduvaud.ch	 :																				
Lieu de travail :	CFPV (COFOP-ETML), Avenue de Valmont 28b, 1010 Lausanne																					
Orientation :	88601 Développement d'application																					
Chef de projet :	Nom : CARREL	Prénom : XAVIER																				
	 : xavier.carrel@eduvaud.ch	 : 079 212 96 21																				
Expert 1 :	Nom : OBERSON	Prénom : Bernard																				
	 : oberson.bernard@gmail.com	 : 078 708 02 96																				
Expert 2 :	Nom : WOLF	Prénom : Benjamin																				
	 : bw-tpi@hotmail.com	 : 024 557 64 48																				
Période de réalisation :	Du <b>lundi 1 mai 2023 à 8h</b> au <b>mercredi 31 mai 2023 à 12h15</b>																					
Horaire de travail :	<table><tr><td>Lundi</td><td>08h00-11h25</td><td>13h10-16h35</td><td><i>Pentecôte 29 mai</i></td></tr><tr><td>Mardi</td><td>-</td><td></td><td></td></tr><tr><td>Mercredi</td><td>08h00-12h15</td><td>13h10-16h35</td><td></td></tr><tr><td>Jeudi</td><td>-</td><td>13h10-16h35</td><td><i>Ascension 18 mai</i></td></tr><tr><td>Vendredi</td><td>08h00-12h15</td><td>13h10-16h35</td><td><i>Pont de l'Ascension 19 mai</i></td></tr></table> <p><i>Toutes les demi-journées ont une pause obligatoire de 15 minutes.</i></p>		Lundi	08h00-11h25	13h10-16h35	<i>Pentecôte 29 mai</i>	Mardi	-			Mercredi	08h00-12h15	13h10-16h35		Jeudi	-	13h10-16h35	<i>Ascension 18 mai</i>	Vendredi	08h00-12h15	13h10-16h35	<i>Pont de l'Ascension 19 mai</i>
Lundi	08h00-11h25	13h10-16h35	<i>Pentecôte 29 mai</i>																			
Mardi	-																					
Mercredi	08h00-12h15	13h10-16h35																				
Jeudi	-	13h10-16h35	<i>Ascension 18 mai</i>																			
Vendredi	08h00-12h15	13h10-16h35	<i>Pont de l'Ascension 19 mai</i>																			
Nombre d'heures :	89 heures																					
Planning (en H ou %)	Analyse 15%, Implémentation 50%, Tests 15%, Documentation 20%																					
Présentation :	Dates retenues : 7 ou 8 juin 2023																					

**2 PROCÉDURE**

Le candidat réalise un travail personnel sur la base d'un cahier des charges reçu le 1er jour.

Le cahier des charges est approuvé par les deux experts. Il est en outre présenté, commenté et discuté avec le candidat. Par sa signature, le candidat accepte le travail proposé.

Le candidat a connaissance de la feuille d'appréciation avant de débiter le travail.

Le candidat est entièrement responsable de la sécurité de ses données.

En cas de problèmes graves, le candidat avertit au plus vite les deux experts et son CdP.

Le candidat a la possibilité d'obtenir de l'aide, mais doit le mentionner dans son dossier.

A la fin du délai imparti pour la réalisation du TPI, le candidat doit transmettre par courrier électronique le dossier de projet aux deux experts et au chef de projet. En parallèle, une copie papier du rapport doit être fournie sans délai en trois exemplaires (L'un des deux experts peut demander à ne recevoir que la version électronique du dossier). Cette dernière doit être en tout point identique à la version électronique.

---

### 3 TITRE

Runeo-Drive

L'application mobile qui accompagne Runeo-Desk, destinée aux chauffeurs d'artiste du Paleo

Festival.

---

### 4 MATÉRIEL ET LOGICIEL À DISPOSITION

- 1 poste de travail ETML
- Suite Office
- IceScrum
- Repository Github ETML-INF/Runeo-Drive contenant le code et la documentation de l'application actuelle.
- Repository Github ETML-INF/Runeo-Desk contenant l'ensemble du backend
- Framework React Native, public et gratuit
- IDE au choix du candidat, public et gratuit
- Le chef de projet est à disposition pour effectuer des adaptations nécessaires identifiées et validées du backend

---

### 5 PRÉREQUIS

- Accès en écriture à ETML-INF/Runeo-Drive
- Accès en lecture à ETML-INF/Runeo-Desk
- Modules de développement Web
- Expérience du framework React Native
- Connaissance des méthodes Agile
- Expérience de l'outil IceScrum
- Expérience de l'outil Swagger

## 6 DESCRIPTIF DU PROJET

L'application Runeo-Drive est la companion-app de Runeo-Desk. Ensemble, ces deux applications permettent au groupe des chauffeurs du Paleo Festival de gérer efficacement les transports des artistes, ainsi que de leur entourage (staff, agent, musiciens, ...).

L'application (React Native) nécessite des améliorations qui sont l'objet de ce projet.

### 6.1 Page de profil

Accédée à partir du sommet de la liste des chauffeurs, cette page montre les informations de profil de l'utilisateur connecté:

- Photo
- Prénom, nom et nom d'affichage (pseudo)
- Email
- Groupe
- Rôle
- Status

### 6.2 Correction de status

Dans le cas où je constate dans ma page de profil que mon status est incorrect, l'application me permet de le mettre à jour au moyen d'une liste déroulante dont les éléments sont fournis par le backend.

### 6.3 Mes horaires et mes runs


Accédée à partir du sommet de la liste des chauffeurs, cette page montre les horaires de mon groupe dans une vue de type calendrier.

Cette vue est continue et verticale, c'est-à-dire qu'on peut scroller verticalement de manière continue pour voir les horaires précédents et suivants.

A l'intérieur de chaque horaire figurent les runs qui me concernent. La maquette ci-contre n'est qu'une suggestion de mise en page possible.

La couleur des runs doit refléter leur état (futur, en cours ou terminé).

Pour chaque run, on a :

- Obligatoirement son numéro
- Obligatoirement une icône « multiple » s'il y a plus d'un véhicule dans le run (p.ex : )
- Le nom de l'artiste si on a la place
- Le nom du run si on a encore de la place
- Le nom du ou des autres chauffeurs s'il y en a et qu'il y a de la place pour les noms

L'application doit montrer les informations obligatoires et autant d'information optionnelle que possible en fonction de l'espace disponible. Elle peut choisir de supprimer ou d'écourter une information optionnelle trop longue.



## 6.4 Gestion d'erreur

L'application est fortement dépendante de l'API offerte par le backend, laquelle peut être parfois indisponible ou ne pas fonctionner correctement.

L'application doit prendre en compte les divers cas d'erreurs possibles (pas de réseau, pas de réponse, code d'erreur http, ...) et les traiter proprement d'un point de vue UX : messages en français, non techniques, timeouts, ...)

## 7 LIVRABLES

Le candidat est responsable de livrer à son chef de projet:



- Une planification initiale (dans le rapport de projet) le lundi 1.05 à 16h35
- Le rapport de projet dans son état actuel, en PDF dans le dossier (ou sous-dossier de) `doc/TPI-Sartoni` du repo, tous les mercredis et vendredis à 16h35
- Le journal de travail dans son état actuel, en PDF dans le dossier (ou sous-dossier de) `doc/TPI-Sartoni` du repo, tous les mercredis et vendredis à 16h35
- Des commits Gits bien formés, selon les directives établies dans le document « Git.pdf » sur Teams
- Une release Github finale le mercredi 31.05 à 12h15, contenant
  - Toutes les sources
  - Le rapport final, en PDF dans le dossier (ou sous-dossier de) `doc/TPI-Sartoni` du repo
  - Le journal de travail final, en PDF dans le dossier (ou sous-dossier de) `doc/TPI-Sartoni` du repo
- Deux copies papier (pour les experts) de tous les documents finaux le mercredi 31.05

## 8 POINTS TECHNIQUES ÉVALUÉS SPÉCIFIQUES AU PROJET

La grille d'évaluation définit les critères généraux selon lesquels le travail du candidat sera évalué (documentation, journal de travail, respect des normes, qualité, ...).

En plus de cela, le travail sera évalué sur les 7 points spécifiques suivants (Point A14 à A20):

1. Qualité des commits Git (nommage, atomicité et granularité), application de gitflow
2. La spécification avec Swagger des points d'API supplémentaires ou modifiés
3. La gestion du changement de status (UX, détection et traitement d'erreur, couverture des cas possibles)
4. Revue de code (évaluée selon les critères vus en classe)
5. Le rendu des horaires + runs (timeline, dimensions, couleurs)
6. La gestion du contenu texte de la box de chaque run (dépassements, stratégie de raccourcissement)
7. La validation sur des plateformes physiques iOS et Android (il n'est pas attendu que l'application soit disponible sur AppStore/GooglePlay)

### Remarque :

Le recours à des outils en ligne d'intelligence artificielle (ex. : Chat GPT) doit être mentionné et ne peut servir que d'inspiration à la réalisation. En cas d'abus, l'évaluation du TPI en tiendra compte.

## 9 VALIDATION

	Lu et approuvé le :	Signature :
--	---------------------	-------------

Candidat :		
Expert n°1 :		
Expert n° 2 :		
Chef de projet :		